# Pick-up Game: Acquainting Neighbors Quickly and Efficiently in Crowd

JunZe Han[†], Xiang-Yang Li[†],[§]
[†]Department of Computer Science, Illinois Institute of Technology, Chicago, IL
[§] School of Software and TNLIST, Tsinghua University
Email: jhan20@hawk.iit.edu, xli@cs.iit.edu

***Abstract*—Effective and energy efficient neighbor discovery protocol is a crucial component for the success of emerging opportunistic encounter-based mobile (ad hoc) networking for social and gaming. In this work, we design P-Game, an effective neighbor discovery protocol to find a group of neighbors. By leveraging a quick estimation of the number of neighbors, we design various protocols for collecting the IDs of these neighboring nodes with the objective of either minimizing the latency of neighbor-discovery process or minimizing the active slots (*i.e.* energy consumption) of each neighbor. We validate PickupGame through rigorous theoretical analysis. When the required acquaintances $k$ is a constant fraction of neighbors, our protocol is proved to have optimum delay $\Theta(k)$ and optimum active slots $\Theta(1)$. When we need to find all neighbors, our protocols achieve tradeoffs in the delay and active slots: one approach has discovery latency $O(k \ln\ln k)$ and active slots $O(\ln\ln k)$. Our evaluations corroborate our theoretical results and show considerable improvement in discovery latency over existing approaches in almost all cases.**

## I. Introduction

Thanks to the popularity of mobile devices and the proximate-based applications on mobile devices, we are able to connect virtual social networking with real world. For example, by leveraging the relationships in social network and the proximate-based application, *e.g.*, Who's Near Me [1] and BlueHoo [2], it is possible to create real-world activities and immediately ask friends nearby to join, eg, Sony's Vita [3], just like a *pick-up game*. However not all applications are able to provide a centralized server to help find nearby users and the users may not be willing to disclose their locations to the server. Thus the first step to implement aforementioned applications is to acquaint people in vicinity quickly and efficiently.

In this work we study neighbor discover problem when an *initiator* wants to find $k$ acquaintances from a group of neighbors (with possibly unknown cardinality) for opportunistic social networking and gaming. The challenge here is to balance the energy consumption and the latency: less wireless communication, although resulting in less energy consumption, may result in larger delay for neighbor discovery, since the devices have less chances to "look around". The neighbor discovery problem has been studied extensively, but the majority of prior protocols focus on enabling two nodes to discover each other [4]–[7], *i.e.*, one-to-one discovery protocols. These protocols when applied to address $k$-neighbors discover problem will often have large latency, as discussed and verified later.

To support opportunistic networking initiated by a user, we design **P-Game** for finding $k$-neighbors with small latency and energy consumption. In protocol **P-Game**, the *initiator* first estimates the number of potential *valid* neighbors (with similar interests) in the crowd. Then based on the estimation, various protocols are designed to either minimize the latency of neighbor discovery process, or minimize the energy consumption of each neighboring node, or to provide Pareto optimum solutions when we cannot optimize both metrics simultaneously. Specially our protocols are designed based on two cases: (1) the number $n$ of existing potential neighbors in the crowd is within constant multiples of the number $k$ of requested neighbors; (2) we want to find all potential neighbors, *i.e.*, $k = n$. Rigorous theoretical analysis shows that our protocol has the optimum discovery latency $\Theta(k)$, optimum active slots $\Theta(1)$ for the first case, and it has discovery latency $O(k \ln\ln k)$ and active slots $O(\ln\ln k)$ for the second case. We further design duty-cycle-based protocols with which nodes are able to work in duty-cycle model. Our extensive evaluations show that our protocols outperform previous protocols significantly in terms of discovery latency and energy consumption.

The rest of the paper is organized as follows. We review the related work in Section II. Section III describes the system model and formulates the problem. In Section IV, we present our protocols and analyze their performance. Section V presents the duty-cycle-based protocol and its performance analysis. Our extensive evaluation results are reported in Section VI. Finally we conclude our work in Section VII.

## II. Related Work

Due to the energy constraint, a number of neighbor discovery protocols are designed based on the duty cycle. For a duty cycle network [8]–[10], the time domain is assumed to be divided into slots of fixed length, and each node wakes up at some slots (called *active slots*) randomly or periodically, and sleeps in the remaining slots. When a node is awake it can transmit or receive at current slot (but not both typically). For two nodes to discover each other, both nodes have to be awake at the same slot. The challenge of those protocols for neighbor discovery in duty cycled networks is to make nodes' active slots overlap as many as possible, while at the same time to reduce the number of active slots and collisions.

Existing neighbor discovery protocols can be categorized into two types: **one-to-one** discovery protocol which is designed for two nodes to discover each other, and **many-to-many** discovery protocol which enables all nodes in a clique to discover the existence of other nodes.

**One-to-One discovery protocol:** Mcglynn *et al.* [4] present a family of probabilistic protocols "birthday protocols", in which, two nodes independently and randomly select several slots to transmit and receive, respectively. Tseng *et al.* [11] present a neighbor discovery protocol based on the concept of quorum. Time intervals are organized as a $n \times n$ array and each host picks one column and one row to send a beacon, such that two hosts have at least two intersecting time intervals.

Dutta *et al.* [5] present an asynchronous neighbor discovery protocol Disco, in which each node picks a pair of primes and the sum of their reciprocals equals the desired duty cycle. Based on the Chinese Remainder Theorem, for every pair of nodes, at least at one slot they will wake up together and discover each other. Kandhalu *et al.* [7] propose a neighbor discovery protocol called U-Connect. In this protocol each node also picks a prime number $p$ and wakes up $\frac{p+1}{2}$ times every $p^2$ slots. They also propose the power-latency product metric to model the trade off between energy consumption and discovery latency and to evaluate the performance of the neighbor discovery protocol. Zhang *et al.* [12] propose an on-demand generic discovery accelerating middleware for existing discovery protocols, by leveraging additional energy budget and knowledge of known neighbors.

Recently Bakht *et al.* [6] propose a novel periodic slot-based neighbor discovery protocol *searchlight*. The node wakes up at a fixed slot in each period which is called the *anchor slot*. The node also wakes up at the *probe slot* which moves around in the period. Since the length of the period is restricted to a power-multiple of the smallest duty cycle, the relative position of the anchor slot of one node is fixed with respect to that of the other node. Thus as the probe slot moves around, it will overlap with the anchor slot of the other node in some period, which provides the upper bound on discovery latency. Searchlight also allows nodes to choose different duty cycles according to its energy conservation level.

If we apply one-to-one discovery protocol directly for the one-to-many discovery problem studied in this work, there is a large latency.

**Many-to-Many discovery protocol:** Compared with one-to-one discovery protocols, many-many discovery protocols need to consider collisions when multiple nodes transmit simultaneously. Vasudevan *et al.* [13] and Jakllari *et al.* [14] propose neighbor discovery protocols for sensor nodes using directional antennas. Vasudevan *et al.* [15] propose an ALOHA-like neighbor discovery algorithm and reduce it to the Coupon Collector's Problem. If only one node transmits at a slot, it will be discovered by all other nodes; otherwise no discovery is made due to collisions or idle slot. Their algorithm does not require node to have knowledge of the number of neighbors. They do not consider the duty-cycles of various nodes.

## III. System Model and Problem Formulation

The neighbor-discovery system studied here consists of a set of mobile devices owned by users. Users or devices join the system randomly with their own unique IDs and form a network by communicating through exchanging packets. Note that packets are sent by broadcasting, thus for two devices to communicate exclusively, they have to know the IDs of each other. We assume that the time domain is divided into slots of fixed length and define a **frame** as a series of continuous time slots and denote the $j$th slot of $i$th frame as $t_{i,j}$. The receiver is able to detect collisions when receiving multiple packets simultaneously. The receiver can also recognize an idle slot when no packet received. A node $u$ is said to be *discovered* by a node $v$ if $v$ has obtained $u$'s ID successfully without collision.

We define the states when the device is transmitting or receiving as *active states* and assume that the device consumes significantly less energy in non-active state. We define $s_t(u)$ as the state variable of node $u$ at slot $t$. If node $u$ is active at slot $t$, then we have $s_t(u) = 1$; otherwise $s_t(u) = 0$ if $u$ is non-active.

We also assume that each user specifies its interests by using an interest vector $x$ consisting of several interests represented by real values and the *interest distance* between two users is defined as the distance between their interest vectors. The actual distance between two interest vectors $x$ and $y$ could be computed using *Euclidean distance*, or *vector similarity*. The neighbor of a node is defined in both *physical distance* and *interest distance*, that is, two users are called *neighbors* if the physical distance between them is short, *e.g.*, within the communication range of each other, and share similar interests, *e.g.*, the distance between their interest vectors is bounded from above by a threshold value.

We further assume that users will not leave the system during the process of neighbor discovery and the user arrivals follow the Poisson process with unknown rate $\lambda$ and the system starts at a global slot 0. We denote the number of nodes that joined the system at time slot $t$ as $a_t$, and the total number of nodes in the system at slot $t$ as $A_t$, *i.e.*, $A_t = A_0 + \sum_{i=1}^{t} a_i$.

Let $N_t(u)$ be the set of neighbors who have been discovered by node $u$ by slot $t$. The problem we study here is to design an effective and efficient protocol by which a user can discover the required number (denoted as $k$ in this work) of neighbors to join the activities that she started. During the process of neighbor discovery, we would like to minimize the duration, *i.e.*, $\arg\min_t\{\| N_t(u) \| \geq k\}$, for searching neighbors and the energy cost, *i.e.*, $\sum_{v \in N_t(u)} \sum_{i=1}^{t} s_i(v)$, for each device participating in the discovery process. Here we assume that the initiator $u$ will be active during the majority slots, and thus, we will not minimize the active slots for $u$. For simplicity of presentation, we assume that there are already protocols in place to compute the interests similarity between two users [16], [17]. Thus, we will skip the step of computing interest similarity when describing our protocols.

## IV. Protocol Design

We refer to the node who is looking for neighbors as the **initiator** and all the other nodes who are ready to be discovered as the **participants**. Given the number $k$ of neighbors requested, the discover protocol first needs to decide when to start the discover process, as the number of existing nodes $A_0$ around the initiator when it arrived may be much less than the requested number $k$. Thus we first need to estimate the number of existing nodes around the initiator.

For neighbor discovery, the initiator has to send its ID to all the participants and the participants in turn send their IDs to the initiator. To avoid highly frequent collisions when sending IDs, we carefully select a certain number of continuous slots, *i.e.*, **frame** and let each participant send its ID randomly in one of slots in a frame.

## A. Protocol Description

Our protocol is divided into two phases. In the first phase the initiator will estimate the number of existing participants (with the goal of having at least $k$ participants as its discovered neighbors) by sending out an ESTIMATE message, in which the initiator will specify the ID of the initiator, the probability $p_a$ with which a participant sends an ACK message, and the size of the frame used for estimating. Each participants needs to send an ACK messages in some slot within a frame to report its existence. In the second phase, after obtaining an estimation of the number of participants, the initiator starts to obtain the IDs of neighbors by sending a DISCOVER message including a new frame size for discovery and the participants send ACK messages within a frame to report their IDs.

## B. Estimating the Number of Neighbors

In the first phase, the initiator estimates the number of existing neighbors. Our method is built upon methods for RFID tag estimation, *e.g.*, [18]. We first briefly introduce the idea of the estimation. In this phase the initiator first broadcasts an ESTIMATE message and then waits for a frame to receive ACK messages from its neighbors. Let $n$ be the number of nodes that will send ACK messages within a frame and each such node sends its ACK message at a randomly selected slot. We define the load factor $\rho$ as $\rho = \frac{n}{m}$, where $m$ is the size of a frame. Let $x_0$, $x_1$, $x_2$ be random variables denoting the number of slots in a frame with 0 ACK message received, 1 ACK message received and multiple ACK messages received respectively. Then the expected values of these variables are $E[x_0] = me^{-\rho}$, $E[x_1] = m\rho e^{-\rho}$, and $E[x_2] = m(1-(1+\rho)e^{-\rho})$ Then we can obtain three estimates on $n$ by solving the above three equations separately with the observed value of $x_0$, $x_1$ and $x_2$. Observe that $x_1$ is not a monotone function of $n$, thus, we will not use $x_1$ to estimate the value $n$. In our design, for simplicity, we use the value of $x_0$ to compute the estimated value of $n$ as $\hat{n} = m\ln(m/x_0)$. The variance $\delta_0$ of estimation is $\delta_0 = n\frac{e^{\rho}-(1+\rho)}{\rho}$. When the load factor $\rho$ is large, with high probability a collision will happen and estimating the number of neighbors by using $x_0$ will result in large variance. Thus given the desired variance $\sigma^2$ of the estimation and the size of a frame, there is an upper bound on the number of neighbors whose size can be estimated.

To deal with the case when the number of existing participants is significantly larger than $k$, the participants send ACK messages with a ceratin probability $p_a$ such that the expected number of participants who will send ACK messages is reduced to $A_t \cdot p_a$ and the initiator is able to estimate the number of existing participants $A_t$ with desired confidence. Our protocol **P-Game** works as follows.

**Step-1:** The initiator broadcasts an ESTIMATE message containing the frame size $m$ and reply probability $p_a = 1$.

**Step-2:** Each participant sends an ACK message randomly at one of the following $m$ slots with probability $p_a$. The initiator estimates the number of existing participants based on the number of idle slots $x_0$.

**Step-3:** If the estimator does not reach the desired confidence, the initiator broadcasts another ESTIMATE message with reply probability $p_a \leftarrow 1/2^{2^j}$ for the $j$-th estimation and repeats the steps 2 and 3.

**Setting the frame size:** Recall that the initiator only needs to discover $k$ neighbors. Thus, it is sufficient that we can estimate $\Theta(k)$ neighbors in one frame. In our protocols, the size of the frame is set to be able to estimate up to $4k$ participants. Let $\sigma^2$ be the desired variance of the estimation. Let $n = 4k$ and $\rho = \frac{4k}{m}$, then by solving the equation $\delta_0 \le 4k\frac{e^{\rho}-(1+\rho)}{\rho} \le \sigma^2$, we get a frame size $m$ that is able to reach desired variance when the number of participants is at most $4k$, *i.e.*, $4k\frac{e^{4k/m}-(1+\frac{4k}{m})}{4k/m} = \sigma^2$. The frame size $m$ is the solution of

$$me^{4k/m} - (m+4k) = \sigma^2.$$

**Lemma 1.** *The frame size $m \simeq \frac{8k^2}{\sigma^2}$. When the variance is $\sigma = \frac{k}{2}$, a constant-sized fame is enough for estimation.*

**Adjusting acking probability:** As the neighboring nodes will send an ACK message with a probability $p_a$, $n = A_0 \cdot p_a$ nodes will be responding initially. Let $\hat{n}$ be the estimate of $n$. If $\hat{n}\frac{e^{\rho}-(1+\rho)}{\rho} > \sigma^2$, then we need to decrease the load factor $\rho$, that is, we need to decrease the probability $p_a$. In our design we set $p_a = 1/2^{2^j}$ for the $j$-th frame, if we cannot reach the desired variance. Under this approach, using about $F = \log\log\frac{A_0}{4k}$ frames, we can have at most $4k$ acking neighbors (but the lower bound could be as small as $\frac{16k^2}{A_0}$). To recover more neighboring nodes, we can perform binary search to find an integer $f \in [2^{F-1}, 2^F]$ such that $\frac{A_0}{2^{f-1}} \ge 4k$ and $\frac{A_0}{2^f} \le 4k$. This binary search will take another $1+\log_2(2^{F-1}) = F$ frames. Thus, the total number of frames needed is $2\log\log\frac{A_0}{4k}$. Then we have the following lemma.

**Lemma 2.** *Suppose $A_0 \gg k$ and we set the acking probability in $j$-th estimate frame as $p_a = 1/2^{2^j}$. The number of slots needed to get about $\Theta(k)$ neighbors participating in the phase-2 is $\Theta(\log\log\frac{A_0}{k})$, w.h.p..*

Obviously, for practical values of $A_0 \gg k$, we claim that the number of total slots needed for having $n$ (where $n \in [2k, 4k]$ *w.h.p.*) nodes participating in the phase-2 of the neighbor discovery process is constant *w.h.p.*.

**Not enough neighbors to start with, "dense arrivals":** We assume that during the process of estimation, the number of participants remains the same asymptotically (as estimating will only take $\Theta(\log\log\frac{A_0}{k})$ slots *w.h.p.*) and $\lambda \le \alpha$ for some known upper bound $\alpha$ on the number of participants arriving in one slot.

Assume that $n_0 < k$ is the number of participants estimated by the initiator at slot $t_0$. The initiator then needs to wait for the arrivals of another $\Theta(k - n_0)$ participants. Note that the inter-arrival time $T$, which is the time interval between two arrivals of the participants, is subject to the exponential distribution. The probability that no arrival within $t$ slots, is $\Pr(T > t) = e^{-\lambda t}$, with expected value $\mathrm{E}[T] = 1/\lambda$. Thus the expected number of slots the initiator needs to wait is $(k - n_0)/\lambda$.

The initiator has to estimate the arrival rate $\lambda$. We estimate the value of $\lambda$ by the maximum likelihood estimate $\hat{\lambda} = \frac{1}{\ell}\sum_{j=1}^{\ell} n_j$, where $n_j$ is the number of participants arrived during slots $\in [t_{j-1}, t_j)$, where $t_j$ is the starting slot for the $j$-th estimation and $t_j$, $j \geq 1$, is computed as follows. Let $\hat{\lambda}_0 = \alpha$ be the initial estimate of $\lambda$. The initiator estimates the arrival time $t_1$ of the $\lceil (k-n_0)/2^{\beta} \rceil$-th participant with $\hat{\lambda}_0$, where $\beta > 1$ is a constant integer. Then the initiator estimates the number of participants $n_1$ at slot $t_1$, calculates the new estimate $\hat{\lambda}_1$ of the arrival rate $\lambda$, and then estimates the arrival time $t_2$ of the $\lceil (k-n_0)/2^{\beta-1} \rceil$ participant with $\hat{\lambda}_1$. Then the initiator starts 2-nd estimate of $\lambda$ at slot $t_2$ by estimating the actual arrivals $n_2$. This process ends until enough number participants have arrived, *i.e.*, $\sum_{j=0}^{l} n_j \geq k$.

**Not enough neighbors to start with, and "sparse arrivals":** When $\lambda < 1$ and $A_0 \ll k$, the initiator can try to obtain IDs of initial $A_0$ neighbors by keeping broadcasting DISCOVERY message until enough number of neighbors are discovered. For a newly joined participant, whenever it receives a DISCOVERY message, it immediately sends an ACK message back to the initiator at one of the subsequent two time slots randomly. As the arrival rate is small, the probability that collisions happen is low and hence the initiator can discover a new participant when it appears in vicinity. Thus in this case the initiator is able to discover all remaining $k - A_0$ neighbors in time $\Theta(k - A_0)$.

## C. Obtaining IDs of $k$ Participants

In phase-2, the participants who did not send ACK messages in the last frame of phase-1 will not participate. For simplicity, let $n$ be the number of participants in the second phase. In our protocol design and analysis, two different cases will be considered.

1) **Constant Fraction of Participants**: Initiator wants to discover $k$ participants among $n$ participants with $a_1 k \leq n \leq a_2 k$, for constants $1 < a_1 < a_2$.
2) **All Participants**: The initiator wants to discover all $n$ participants, *i.e.*, $k = n$.

At the beginning of phase-2, the initiator will send a DISCOVER message to all participants with the frame size $m$ (which will be different from the estimate-frame size) and each participant will send its ID in one of $m$ slots in the frame. The initiator will receive ACK messages when no collision happens. We design two different approaches for obtaining ID in phase-2 by choosing different frame sizes with different considerations. Figure 1 presents the timeline of phase-2.

*a) First Approach—P-Game-1:* For the **first** approach, following each slot when the initiator discovers a neighbor, it will send a CONFIRM messages containing the ID obtained at the previous slot; accordingly after sending an ACK message, each participant will stay active at the next slot to wait for the CONFIRM message. If a participant successfully receives the CONFIRM message with its own ID, the participant will not send ACK message in subsequent slots. The following summarizes the basic steps of our first approach for neighbor discovery.

**Step-1:** The initiator broadcasts a message DISCOVER including the frame size $m$.

**Step-2:** The participant sends an ACK message randomly at one of **even** slots within the frame.

**Step-3:** The initiator sends a CONFIRM message at **odd** slots with the ID of the participant discovered, when an ACK message was received successfully from the previous **even** slots.

**Step-4:** If the initiator receives no ACK messages from participants or it has obtained $k$ IDs from participants, it will send END messages to all participants; otherwise step 1-3 will be repeated.

*b) Second Approach—P-Game-2:* In the **second** approach, the initiator does not send CONFIRM message, but only sends one END message after that it has discovered enough neighbors. Thus the basic steps of the second approach consist of only step 1-2, 4 of the first approach; and in step 2 the participants can send ACK messages at any slots instead of at only even slots, and in step 4 the initiator will send END messages when it has obtained $k$ IDs from participants.

## D. Performance Analysis

In this section we will analyze the performance of the two approaches in terms of the discovery latency and the active slots for the participants. The following lower-bound lemma is straightforward:

**Lemma 3.** *For any neighbor discover protocol, the latency is $\Omega(k)$ slots for discovering $k$ neighbors and each participant needs to wake up at least two slots during the process of neighbor discovery.*

*1) Analysis of P-Game-1:* We only consider the slots used to receive ACK messages as the discover latency is at most twice of the number of slots used to receive ACK messages. We analyze the performance of the first approach based on two cases: 1) only the IDs of a constant fraction of all participants are needed, and 2) the IDs of all participants are needed. Our analysis will use the Hoeffding inequality.

**Theorem 1.** ***(Hoeffding).*** *Let $X_1, \cdots, X_n$ be independent random variables and $\Pr(X_i \in [a_i, b_i]) = 1$, $\forall i \in [1, n]$. Let $S = \sum_{i=1}^{n} X_i$. Then, for $t > 0$, the probability*

$$\Pr(S - E[S] \leq t) \leq \exp(-\frac{2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2})$$

**Corollary 1.** *Let $Y \sim B(n,p)$ be the binomial variable. For $k \leq np$, we have*

$$\Pr(Y \leq k) \leq \frac{1}{2}\exp(-\frac{2(np-k)^2}{n})$$

*a) Constant Fraction of Participants:* Let **F** be the number of frames the initiator needs to discover $k$ neighbors. We first prove an upper bound on **F**.

**Lemma 4.** *Let **F** be the number of frames needed to discover $k$ neighbors. In P-Game-1, when $m \geq n$, and $n \in [a_1 k, a_2 k]$, where $1 < a_1 < a_2$, for some constant $\beta \geq \ln \frac{a_1(1+\epsilon)}{a_1 - 1} / \ln \frac{e}{e-1}$ and $0 < \epsilon < 1$, we have*

$$\Pr(\boldsymbol{F} > \beta) \leq \frac{1}{2}\exp(\frac{-2\epsilon^2}{a_1}k)$$

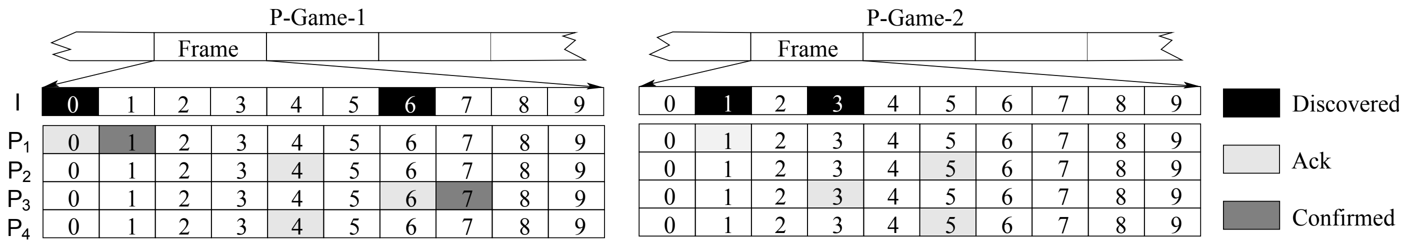


Fig. 1: Neighbor Discovery Process. In P-Game-1, $P_1$ and $P_3$ are discovered at the 0-st and the 6-th slot; and confirmed at the 1-st and the 7-th slot. In P-Game-2, $P_1$ and $P_3$ are discovered at the 1-st and the 3-rd slot

*Proof:* Let $u_i$ be the number of participants whose IDs have not been obtained at the beginning of the $i$-th frame. For a given participant, the probability that its ID is obtained in the $i$-th frame is $\binom{m}{1}\frac{1}{m}(1-\frac{1}{m})^{u_i-1} = (1-\frac{1}{m})^{u_i-1} \geq e^{-1}$. This inequality is due to $u_i \leq n \leq m$. Then we have

$$u_{i+1} = u_i(1-(1-\frac{1}{m})^{u_i-1}) < n(1-\frac{1}{e})^i.$$

We define a binomial variable $S'_q \sim B(n, 1-(1-1/e)^q)$. As $n \in [a_1k, a_2k]$ for some constants $1 < a_1 < a_2$, when $q \geq \ln\frac{a_1(1+\epsilon)}{a_1-1} / \ln\frac{e}{e-1}$, we have $n(1-(1-1/e)^q) \geq k(1+\epsilon)$. Then according to Corollary 1, the probability that $S'_q \leq k$, where $q = \ln\frac{a_1(1+\epsilon)}{a_1-1} / \ln\frac{e}{e-1}$ and $n \in [a_1k, a_2k]$, is $\Pr(S'_q \leq k) \leq \frac{1}{2}\exp(-\frac{2(n(1-(1-1/e)^q)-k)^2}{n}) \leq \frac{1}{2}\exp(\frac{-2\epsilon^2}{a_1}k)$.

Let $S_q$ be the number of participants discovered by the $q$-th frame. As the probability that a given participant is discovered by the $q$-th frame is larger than $1-(1-1/e)^q$, we have $\Pr(S_q \leq k) < \Pr(S'_q \leq k)$. The event $\mathbf{F} > q$ and the event $S_q < k$ are equal and hence $\Pr(\mathbf{F} > q) = \Pr(S_q \leq k)$. Let $q = \ln\frac{a_1(1+\epsilon)}{a_1-1} / \ln\frac{e}{e-1}$, then we have $\Pr(\mathbf{F} > q) \leq \frac{1}{2}\exp(\frac{-2\epsilon^2}{a_1}k)$. This finishes the proof. ■

Consequently, the process of neighbor discover will be completed within $O(1)$ frames with high probability when discovering a fraction of the existing participants. In this case, the best choice of frame-size is $m = \Theta(n)$ for reducing the discovery latency. Thus, the discovery latency $\mathcal{D}$ is $\Theta(k)$. Since a participant only wakes up twice in each frame, we have

**Lemma 5.** *In P-Game-1, each participant needs to wake-up for $O(1)$ slots when $n \in [a_1k, a_2k]$ for some constants $1 < a_1 < a_2$, and frames size $m \geq n$.*

*b) All Participants:* We then analyze the performance when the initiator wants to discover the IDs of all the participants, *i.e.*, $n = k$.

**Lemma 6.** *In P-Game-1, the optimal frame size is $m = n$ when $n = k$, and the latency for discovering all $k$ neighbors when $n = k$ is $O(n \ln\ln n)$ w.h.p..*

*Proof:* We approximate the neighbor discovery process by dividing it into *epochs* consisting of several frames as shown in Figure 2. We assume that at the beginning of the $i$-th epoch there are at most $W_i = \alpha_i\beta_i n$ participants who have not been discovered, where

$$\alpha_i = \alpha_1^{2^i}, \quad \beta_i = (\frac{n}{m})^{2^i-1},$$

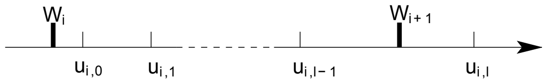


Fig. 2: Neighbor discovery timeline: $[W_i, W_{i+1})$ is the $i$-th epoch. There are $u_{i,0}$ undiscovered neighbors for the first frame in the $i$-th epoch, with $u_{i,0} \geq W_i$. Epoch $i$ has $l$ frames.

for some constant $0 < \alpha_1 < 1$. For simplicity of analysis, in the rest of the proof, we will remove the last frame from each epoch (*i.e.*, frame from $u_{i,l-1}$ to $u_{i,l}$ undiscovered neighbors in Fig. 2), which is then called modified-epoch. Thus, at the end of the $i$-th modified-epoch there are at least $W_{i+1}$ participants not discovered. We assume that the CONFIRM message is sent at the end of the epoch and the number of participants sending ACK messages remains the same in one epoch. Clearly, the discovery latency using this modified-epoch approach is at least the latency of P-Game-1. We will show that each modified-epoch is consisted of a constant number of frames and the total number of epoches is $O(\ln\ln n)$ *w.h.p.*.

Let $u_j$ be the number of participants who have not received the CONFIRM messages by the $j$-th frame. According to the classical ball and bin problem, we know that when $u_j \leq m^{1/2}$, all the remaining participants can be discovered in a constant number of frames *w.h.p.*. Thus, as long as $W_i \leq m^{1/2}$, the remaining participants can be discovered using a constant number of frames. Then the number of epoches is $\lceil\log(\frac{\ln m}{2\ln\frac{m}{\alpha_1 n}})\rceil$. Note when $m = \Omega(n^2)$, the discovery process will be finished in $O(1)$ frames. Thus, an upper bound on the frame-size is $m = O(n^2)$. Consequently, the number of epoches needed is $\leq \log(\frac{\ln n}{2\ln(1/\alpha_1)}) = O(\ln\ln n)$ regardless of frame-size $m$.

We then show that every modified-epoch is consisted of a constant frames *w.h.p.* for all modified-epoches. Consider the $i$-th modified-epoch. Let $u_{i,0}$ be the total number of undiscovered participants at the beginning of the epoch. The probability that a given participant is not discovered in one frame when $W_i \geq u_{i,0} \geq W_{i+1}$ is less than

$$1 - e^{-u_{i,0}/m} \leq u_{i,0}/m \leq \alpha_i\beta_i n/m.$$

Let $u_{i,l}$ be the actual number of participants whose ack have not been received successfully by the $l$-th frame in the $i$-th modified-epoch. Then it is easy to show that, when $l \geq 2$,

$$u_{i,l} \leq u_{i,0}(\alpha_i\beta_i n/m)^l \leq \alpha_1 W_{i+1}$$

Let $S_i \sim B(u_{i,0}, 1-(\alpha_i\beta_i n/m)^l)$ be the number of participants discovered during the first $l$ frames of the $i$-th epoch, where $u_{i,0} \geq m^{1/2}$. Then according to Corollary 1, the probability that $S_i \leq u_{i,0} - W_{i+1}$ is

$$\Pr(S_i \leq u_{i,0} - W_{i+1}) \leq \frac{1}{2}\exp(-2(1-\alpha_1)^2 m^{1/2}).$$

Thus the probability that $l > 2$ is at most $\frac{1}{2}\exp(-2(1-\alpha_1)^2 m^{1/2})$. As the number of epoches is $\leq \log(\frac{\ln n}{2\ln(1/\alpha_1)})$, the probability that total number of frames in all epoches is at most $3\log\frac{\ln n}{2\ln(1/\alpha_1)} = O(\ln\ln n)$ is at least

$$1 - \sum_{\log\frac{\ln n}{2\ln(1/\alpha_1)}} \frac{1}{2}\exp(-2(1-\alpha_1)^2 m^{1/2})$$
$$= 1 - \log\frac{\ln n}{2\ln(1/\alpha_1)}\exp(-2(1-\alpha_1)^2 m^{1/2})/2$$

That is all neighbors can be discovered within $\Theta(\ln\ln n)$ frames *w.h.p.*. We next will determine the size of the frame $m$ to reduce the discovery latency $\mathcal{D}$:

$$\mathcal{D} = m\log(\frac{\ln m}{2\ln\frac{m}{\alpha_1 n}})$$

When $m = n$, we have $\mathcal{D} = O(n\ln\ln n)$. Thus we only need to consider the frame size of $n \leq m < n\ln\ln n$ (otherwise, the latency is at least $m \geq n\ln\ln n$ when $m \geq n\ln\ln n$). Note that, when $n \in [n, n\ln\ln n]$, the discovery latency $\mathcal{D} > n\log(\frac{\ln n}{2\ln\frac{n\ln\ln n}{\alpha_1 n}}) = \Omega(n\ln\ln n)$. Thus, the optimal frame size for reducing the latency is $m = n$. The minimum discovery latency is $O(n\ln\ln n)$. ■

Similarly, we can show the following lower-bound on the number of frames and the discovery latency.

**Lemma 7.** *In P-Game-1, when $n = k$ the latency for discovering all $k$ neighbors is $\Omega(n\ln\ln n)$ w.h.p..*

Notice that a participant only stays active for two slots in each frame. Consequently, we have

**Lemma 8.** *In P-Game-1, the participant needs to stay active for $O(\ln\ln n)$ slots w.h.p. when $n = k$.*

*2) Analysis of P-Game-2:* Let $\mathbf{F}$ be the number of frames the initiator needs to obtain IDs of $k$ neighbors. Similar to the proof of Lemma 4, $\mathbf{F}$ is a constant when $m \geq n$ and $n \in [a_1 k, a_2 k]$, and the discovery latency is $O(m)$. Thus the optimal size of the frame for P-Game-2 is also $m = n$ when $n \in [a_1 k, a_2 k]$.

**Lemma 9.** *In P-Game-2, when $n \in [a_1 k, a_2 k]$ for constants $1 < a_1 < a_2$, the optimal frame size is $m = n$ and the latency $\mathcal{D}$ to discover $k$ neighbors is $O(k)$ w.h.p..*

We next will analyze the performance when $n = k$. In P-Game-2 without CONFIRM message, since the participant wakes up at most once in every frame, the number of slots at which the participant wakes up equals to the total number of frames of discovery process. Thus we have

**Lemma 10.** *In P-Game-2, the number of frames $\mathbf{F}$ (also the active slots of a participant) needed to discover all $n$ participants with probability at least $1 - n^{-\beta+1}$ with frame size $m = n$, $m = n\ln n$ and $m = n^2$ for constant $\beta > 1$ is*

$$\begin{cases} \frac{\beta\ln n}{\ln(e/(e-1))} & \text{if } m = n, \\ \beta\ln n/\ln\ln n & \text{if } m = n\ln n, \\ \beta & \text{if } m = n^2, \end{cases}$$

*Proof:* The probability that a participant's ID is obtained in one frame is $\binom{m}{1}\frac{1}{m}(1-\frac{1}{m})^{n-1} = (1-\frac{1}{m})^{n-1} \geq e^{-n/m}$. The probability that a neighbor has not been discovered by the $i$-th frame is less than $(1-e^{-n/m})^i$ and the probability that not all neighbors are discovered is less than $n(1-e^{-n/m})^i$.

**Case-1.** If $m = n$, then the probability that not all neighbors are discovered is less than $n(1-e^{-1})^i$. It is easy to prove that $\Pr(\mathbf{F} > \frac{\beta\ln n}{\ln(e/(e-1))}) \leq n^{-\beta+1}$.

**Case-2.** If $m > n$, since $1 - e^{-n/m} \leq \frac{n}{m}$, the probability that not all neighbors are discovered by the $i$-th frame is less than $n(1-e^{-n/m})^i \leq \frac{n^{i+1}}{m^i}$. Thus, we have $\Pr(\mathbf{F} > i) \leq \frac{n^{i+1}}{m^i}$.

**If** $m = n\ln n$ and $i = \beta\ln n/\ln\ln n$, then we can prove that $\Pr(\mathbf{F} > \beta\ln n/\ln\ln n) \leq \frac{n^{i+1}}{(n\ln n)^i} = \frac{n}{(\ln n)^i} \leq n^{-\beta+1}$.

**If** $m = n^2$ and $i = \beta > 0$ then we have $\Pr(\mathbf{F} > i) \leq \frac{n^{i+1}}{n^{2i}} = n^{-i+1} = n^{-\beta+1}$.

Notice when $m > n^2$, the initiator only needs constant frames to discover all neighbors. Thus $\Theta(n^2)$ is upper bound on the frame-size. ■

Similarly, we can bound the discover latency as

**Lemma 11.** *In P-Game-2, the latency to discover all $n$ participants w.h.p. with frame size $m = n$, $m = n\ln$ and $m = n^2$ is*

$$\begin{cases} \frac{\beta n\ln n}{\ln(e/(e-1))} & \text{if } m = n, \\ \beta n\ln^2 n/\ln\ln n & \text{if } m = n\ln n, \\ \beta n^2 & \text{if } m = n^2 \end{cases}$$

## V. Duty-Cycle-Based Protocol

In this section, we design neighbor discovery protocol when participants work in a duty-cycle fashion. For simplicity of analysis, we first assume that all participants will adopt the same period $p$.

As the length of a period $p$ does not depend on the number of participants $n$, when $p < n$ there will be a large number of collisions in a period. In order to reduce the collisions resulted from short duty-cycle period, the initiator will broadcast a DISCOVERY message containing the estimated number $n$ of the participants. Specially if a participant's period $p$ is larger than the number of participants $n$, then it sends an ACK message normally when it is awake. However the participant with period $p < n$ only sends an ACK message every $\lceil n/p \rceil$ periods when awake, though it wakes up once every period.

When $n \in [a_1 k, a_2 k]$, we can assume that there is a **virtual frame** of size $p$ if $p \geq n$; or of size $n$ if $p < n$. Then similar to the proof of Lemma 4 and Lemma 9, we have the following lemma.

**Lemma 12.** *When all $n$ participants have the same duty-cycle period $p$, the latency to discover $k$ neighbors w.h.p. is $O(\max(n,p))$ if $n \in [a_1k, a_2k]$ for constants $1 < a_1 < a_2$.*

For the case when the initiator needs to discover all participants, *i.e.*, $n = k$, we have the following lemma.

**Lemma 13.** *When all $n$ participants have the same duty-cycle period $p$, the latency to discover all $n$ participants w.h.p. is $O(\max(p,n)\ln n)$.*

*Proof:* Here we can assume that there is a **virtual frame** of size $m = \max(p,n)$ for the participant. As $m \geq n$ the probability that not all neighbors are discovered by the $i$-th virtual frame is less than $n(1-e^{-1})^i$. Then it is easy to prove that with $\frac{\beta \ln n}{\ln(e/(e-1))}$ frames, the probability that not all neighbors are discovered is less than $n^{-\beta+1}$. Thus, the total number of slots needed is $\frac{\beta \max(p,n) \ln n}{\ln(e/(e-1))}$. ■

We next will analyze the case when the $n$ participants have different duty-cycle periods $p_1 \leq p_2 \leq \cdots \leq p_n$ and $p_n > n$. Note that when $p_n \leq n$, the size of virtual frame is $n$ and the analysis is the same as the case when all participants have the same period $p \leq n$.

**Lemma 14.** *When the length of the largest duty-cycle period $p_n > n$, the latency to discover $k$ participants w.h.p. is $O(\max(k, p_k))$ when the number of participants $n \in [a_1k, a_2k]$ for some constants $1 < a_1 < a_2$.*

*Proof:* We will consider two cases: $p_k \leq n$ and $p_k > n$, *i.e.*, whether there are $k$ participants with duty cycle period less or equal than $n$.

For the first case when $p_k \leq n$, there are $k$ participants with duty-cycle period less or equal than $n$ and each of them will send one ACK message every $n$ slots. Then according to the proof of Lemma 4, we can show that the initiator needs $O(k)$ slots to discover requested $k$ neighbors *w.h.p.*.

For the second case when $p_k > n$, we can consider the size of virtual frame as $p_k$, then each participant sends at most $\lceil p_k/n \rceil$ ACK messages every $p_k$ slots. Thus the total number of ACK messages sent from $n$ participants is less than $n\lceil p_k/n \rceil$ in every $p_k$ slots. Let the size of the virtual frame be $p_k$, then similar to the proof of Lemma 4, the total number of slots needed for the initiator to discover $k$ neighbors is $O(p_k)$. ■

For the case when the initiator needs to discover all participants and $p_k > k$, we can consider the size of the virtual frame as $p_k$. Then similar to the proof of Lemma 13, we have

**Lemma 15.** *When the length of the largest duty-cycle period $p_k > k$, the latency to discover all $n$ participants w.h.p. is $O(p_k \ln k)$ where $n = k$.*

## VI. Evaluation

In this section, we evaluate the performance of our protocols and present the simulation results. Specially, we evaluate the discovery latency and the energy consumption, *i.e.*, number of active slots during the neighbor discovery process. As the actual discovery latency depends on the length of the slot which is dependent on the transmission module, we evaluate the discovery latency in terms of the total number of slots of the discover process. Since the device consumes much more energy when communicating in active state than that in non-active state, we use the number of active slots as the metric for the energy consumption.

We also compare our duty-cycle-based protocol with Searchlight at the same duty cycle. Searchlight has two versions: Searchlight-S and Searchlight-R. For Searchlight-S with sequential probing, the probe slot moves forward one slot every period, thus if two participants choose the same probing sequence, then they cannot be discovered by the initiator due to the collisions. Thus we compare our protocol with Searchlight-R with randomized probing, in which each participant randomly chooses its probing sequence. Since each participant wakes up twice in a period of length $p$, the duty cycle of Searchlight is $2/p$.

### A. Estimate the Number of Neighbors

We first evaluate the performance of our protocol in phase-1 for estimation. In this set of experiments, we first look at the latency of estimating the number of neighbors when there are enough neighbors nearby. We set the frame size such that the estimation variance $\sigma^2$ is less than $k/2$ and test the estimation latency for different requested numbers of neighbors $k$ and different numbers of existing neighbors $n_0$. As shown in Figure 3, the estimation latency does not keep increasing with the number of neighbors, which means our protocol can quickly estimate the number of neighbors when there exists a large number of neighbors nearby. We also look at the number of neighbors that have sent ACK messages in the last frame of Phase-1 and will participate in the Phase-2. In order to avoid too much collisions in Phase-2, it is desirable that the number of participants in Phase-2 is close to the requested number of neighbors $k$. Figure 4 shows the cumulative distribution of the number of participants $n$ in Phase-2 with requested number $k = 20$ and different numbers of existing neighbors $n_0$ in Phase-1. As the figure shows, the number of participants $n$ in Phase-2 increases with the initial number of neighbors $n_0$, while the number $n$ is restricted within $[20, 80]$, *i.e.*, $[k, 4k]$.

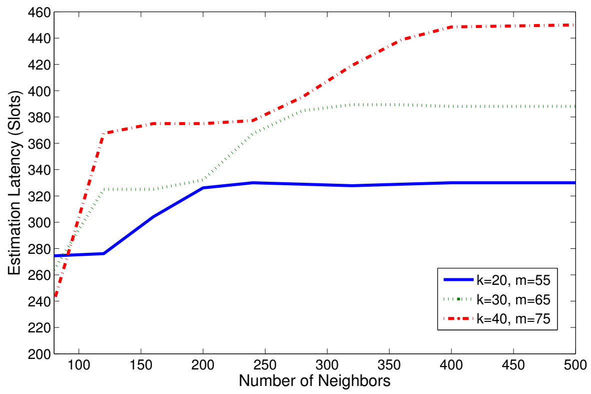


Fig. 3: Estimation latency in Phase-1

We then evaluate our approach for estimating the arrival rate when there are not enough neighbors nearby at the beginning of Phase-1. In this set of experiments, we assume that

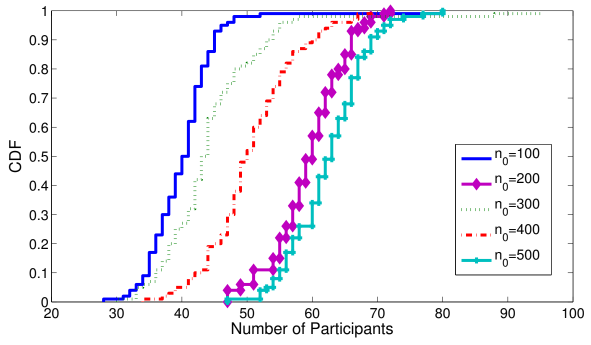


Fig. 4: CDF of participants who sent ACK messages in the last frame of Phase-1 (k=20)

the number of nodes arriving is subject to Poission distribution and we test our approach with different arrival rates. Since we estimate the number of arrived neighbors in discrete steps, our approach is designed such that the initiator can start Phase-2 as soon as enough nodes have arrived, *i.e.*, the number of arrived neighbors is close to the requested number $k$. Figure 5 shows the the number of arrived nodes at the last estimation when Phase-1 ends, where the initial number of nodes is 10 and the requested number is 40. For 90% cases in our experiments when the number of arrived nodes reaches 90 ($< 3k = 120$), the initiator will start Phase-2.

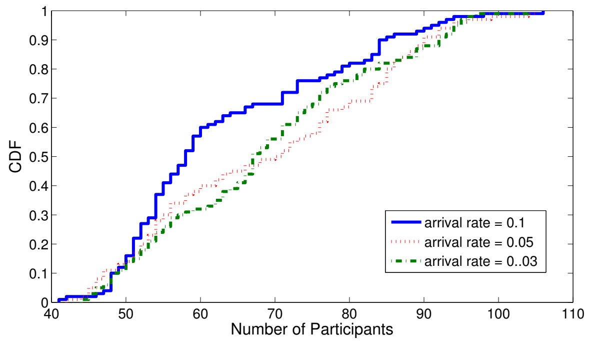
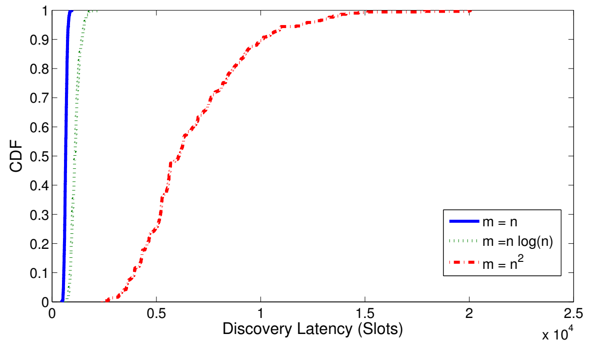


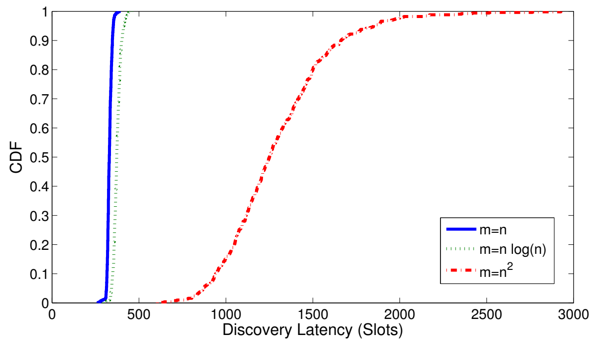
Fig. 5: CDF of participants at the last estimation in Phase-1 when there are not enough participants at the first estimation ($n_0 = 10$, $k = 40$)

## B. Discovery Latency and Active Slots

In this set of experiments we first study how the frame size has impact on the discovery latency. We set the frame size $m$ as $m = n$, $m = n\log(n)$ and $m = n^2$, respectively. As shown in Figure 6a and Figure 6b, the discovery latency increases with the size of the frame for both P-Game-1 and P-Game-2, which is consistent with our theoretical analysis. Since P-Game-1 requires extra slots for confirmation, the discovery latency of P-Game-1 is larger than that of P-Game-2 as shown in figures.



(a) CDF of discovery Latency with P-Game-1 ($k = 20$, $n_0 = 100$)



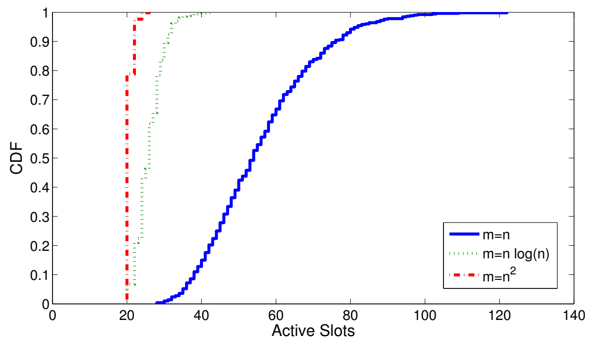
(b) CDF of discovery latency with P-Game-2 ($k = 20$, $n_0 = 100$)

Fig. 6: Discovery Latency

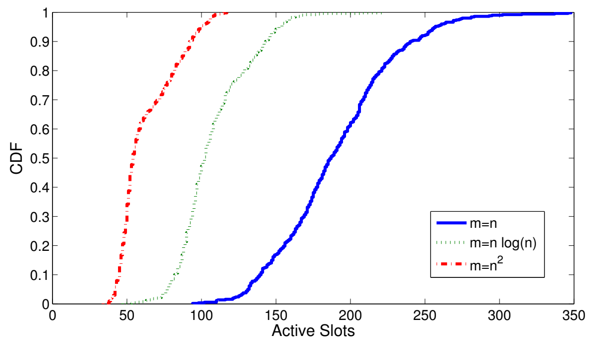
We then evaluate the energy consumption in terms of the total active slots of all the participants. Figure 7a and Figure 7b show the CDF of the total number of active slots. As shown in the figures, the number of active slots reduces with the frame size which is also consistent with our theoretical analysis. The reason is that the collision rate becomes lower when the frame size is larger and hence with higher probability that a participant can be discovered in a frame. Comparing P-Game-1 and P-Game-2, since participants that have been confirmed will not be active in the rest of the neighbor discovery process in P-Game-1, the total number of active slots of P-Game-1 is less than that of P-Game-2.

## C. Duty-Cycle-Based Protocol

We compare the discovery latency of our duty-cycle-based protocol (5% duty-cycle) with Searchlight-R which achieves the best performance among several neighbor discovery protocols [6]. For Searchlight-R, it uses the period of 40 slots such that the duty cycle is 5%. As shown in Figure 8, it takes more slots for Searchlight-R to discover requested number of neighbors than our protocols does with different numbers of participants. The reason our protocol performs better is that the Searchlight is originally designed for a pair of nodes to discover each other, so they do not deal with the collisions when multiple participants sending messages simultaneously. Thus for Searchlight-R when the number of participants is large, the initiator needs to spend more slots to discover neighbors due to the high collision rate.



(a) CDF of total active slots with P-Game-1 ($k = 20$, $n_0 = 100$)



(b) CDF of total active slots with P-Game-2 ($k = 20$, $n_0 = 100$)

Fig. 7: Total Active slots

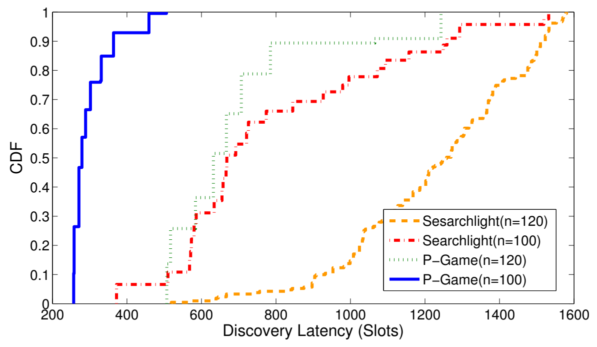


Fig. 8: CDF of discovery latency with P-Game and Searchlight($k = 20$, $5\%$ duty cycle)

## VII. Conclusion

In this work, we study the neighbor discovery problem for opportunistic networking and design efficient one-to-many neighbor discovery protocols. There are several interesting questions left for future research. In this work the user searches neighbors only in one-hop and neighbors have no knowledge of each other. We would like to design protocols for multi-hop neighbor discovery where we need to build a connected network, by exploiting partial knowedge of each neighbor. We also assumed that, during the process of neighbor discovery, the initiator has to quit the duty-cycle model and keeps staying at active state. As a future work we would like to enable the initiator to work in duty-cycle model when searching for neighbors.

## References

[1] "Whos near me," http://www.synergetechsolutions.com/whos-nearme.aspx. [Online]. Available: http://www.synergetechsolutions.com/whos-nearme.aspx.

[2] "Bluehoo," http://bluehoo.com. [Online]. Available: http://bluehoo.com.

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

[4] M. McGlynn and S. Borbash, "Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks," in *Mobihoc*. ACM, 2001, pp. 137–145.

[5] P. Dutta and D. Culler, "Practical asynchronous neighbor discovery and rendezvous for mobile sensing applications," in *Sensys*. ACM, 2008, pp. 71–84.

[6] M. Bakht, M. Trower, and R. Kravets, "Searchlight: won't you be my neighbor?" in *Mobicom*. ACM, 2012, pp. 185–196.

[7] A. Kandhalu, K. Lakshmanan, and R. Rajkumar, "U-connect: a low-latency energy-efficient asynchronous neighbor discovery protocol," in *IPSN*. ACM, 2010, pp. 350–361.

[8] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Sensys*. ACM, 2004, pp. 95–107.

[9] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient mac protocol for wireless sensor networks," in *INFOCOM*, vol. 3. IEEE, 2002, pp. 1567–1576.

[10] M. Buettner, G. Yee, E. Anderson, and R. Han, "X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks," in *Sensys*. ACM, 2006, pp. 307–320.

[11] Y. Tseng, C. Hsu, and T. Hsieh, "Power-saving protocols for ieee 802.11-based multi-hop ad hoc networks," in *INFOCOM*, vol. 1. IEEE, 2002, pp. 200–209.

[12] D. Zhang, T. He, Y. Liu, Y. Gu, F. Ye, R. K. Ganti, and H. Lei, "Acc: generic on-demand accelerations for neighbor discovery in mobile applications," in *Sensys*, 2012, pp. 169–182.

[13] S. Vasudevan, J. Kurose, and D. Towsley, "On neighbor discovery in wireless networks with directional antennas," in *INFOCOM*, vol. 4. IEEE, 2005, pp. 2502–2512.

[14] G. Jakllari, W. Luo, and S. Krishnamurthy, "An integrated neighbor discovery and mac protocol for ad hoc networks using directional antennas," *Wireless Communications, IEEE Transactions on*, vol. 6, no. 3, pp. 1114–1024, 2007.

[15] S. Vasudevan, D. Towsley, D. Goeckel, and R. Khalili, "Neighbor discovery in wireless networks and the coupon collector's problem," in *Mobicom*. ACM, 2009, pp. 181–192.

[16] E. Spertus, M. Sahami, and O. Buyukkokten, "Evaluating similarity measures: a large-scale study in the orkut social network," in *KDD*. ACM, 2005, pp. 678–684.

[17] C. Strecha, A. M. Bronstein, M. M. Bronstein, and P. Fua, "Ldahash: Improved matching with smaller descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, pp. 66–78, 2012.

[18] M. Kodialam and T. Nandagopal, "Fast and reliable estimation schemes in rfid systems," in *Mobicom*. ACM, 2006, pp. 322–333.