A Novel Neighbor Discovery Protocol for Ultraviolet Wireless Networks

Leijie Wang and Yiyang Li
Dept. of Electrical Engr.
University of California
Riverside, CA 92521, USA
{lewang,yiyli}@ee.ucr.edu

Zhengyuan Xu
Dept. of Electronic Engr.
Tsinghua University
Beijing 100084, P.R. China
xuzy@tsinghua.edu.cn

Srikanth V. Krishnamurthy
Dept. of Computer Science and Engr.
University of California
Riverside, CA 92521, USA
krish@cs.ucr.edu

ABSTRACT

Ultraviolet (UV) communication is an attractive option for tactical networks or environmental monitoring. The underlying UV PHY layer has unique characteristics that render previously proposed higher layer protocols for RF communications inappropriate or inefficient. Neighbor discovery is an important functional component of a UV ad hoc wireless network. While there has been some work in UV PHY layers, there is very limited work in network study. In this paper, we propose a new neighbor discovery protocol for this setting; unlike prior protocols, our approach alleviates the negative effects of random access based collisions by choosing a leader that arbitrates the discovery process. Without prior knowledge of the number of nodes in the network, the approach facilitates neighbor discovery in a fast, fair and efficient manner. We perform extensive simulations with a realistic UV PHY layer and demonstrate that the approach reduces the required neighbor discovery time by as much as 90 %. We also examine the impact of various system parameters that can be especially useful to UV network and system designers.

Categories and Subject Descriptors

C.2.5 [**Computer-Communication Networks**]: Local and Sensor Networks

General Terms

Design, Performance

Keywords

Neighbor discovery, optical wireless communications, ultraviolet

1. INTRODUCTION

Ultraviolet or UV communications are especially attractive in applications such as environment monitoring and tactical battlefield communications [1]. Recently, there has been significant progress in terms of building advanced low cost, low power and small size light emitting diodes (LEDs) that operate in the deep UV part of the spectrum. Further, high gain photomultiplier tubes (PMTs) that can be used as receivers have contributed to the rapidly increased interest towards the growth of UV communications. Atmospheric scattering allows the establishment of non-line of sight (NLOS) links with UV; UV photons are scattered multiple times between transmitters and receivers. Thus, even if there is no direct line-of-sight (LOS) path, communications can be established. This feature provides significant advantages for the establishment of communications in various complex scenarios where LOS communications are infeasible. Other properties that make UV communications attractive are that a UV link is resistant to RF jamming, and the probability of the signals being intercepted is very low at long distances due to the unique power decay profile of UV signals.

In the applications envisioned, nodes are likely to be deployed in an ad hoc manner. A node is unlikely to have a priori information about its neighbors or the network topology. As discussed later, the problem of neighbor discovery is non-trivial since UV communications are likely to use directional transmissions; the node should realize not only the identities of its neighbors but also the best direction in which to transmit in order to effectively communicate with each such neighbor. Line-of-sight UV signals are prone to blockage by obstacles; fortunately, nodes can establish possibly multiple NLOS links using which they can communicate.

Our objective in this paper, is to design and evaluate an efficient neighbor discovery protocol that (a) allows the fast and efficient discovery of neighbors, (b) can be applied without any prior knowledge of the network topology or configuration, and (c) allows each node to determine the various directions that can be used to communicate with each of its neighbors and *rank* these directions in terms of their effectiveness.

While there have been prior efforts on neighbor discovery for RF wireless networks, they cannot be directly applied in the UV context. First, the unique UV characteristics are different from the propagation effects experienced on an RF channel. Many of the prior approaches (discussed later) rely on GPS or other enhancements which are unlikely to be available in the application scenarios of interest. Secondly, almost all of the previous approaches only discover neighbors using LOS links; neither do they consider NLOS links nor do they provide any assessment on the relative goodness of these links. There has been very limited work on neighbor discovery in the UV context [7]. However, this approach takes significant time to operate and is inefficient since it uses random access as the basis for neighbor discovery. We propose a much more efficient protocol that can speed up the neighbor discovery process dramatically.

The rest of the paper is organized as follows. We present related work on neighbor discovery in Section 2. In Section 3, we intro-

MSWiM'11, October 31–November 4, 2011, Miami, Florida, USA.

duce the propagation characteristics of UV communications. We review the credit-collection based neighbor discovery protocol proposed in [7] in Section 4 to motivate design of our new protocol; we then describe our basic protocol design based on the use of a leader to arbitrate neighbor discovery. In Section 5, we consider unfairness aspects of the neighbor discovery procedure suffered and propose modifications to improve the fairness. In Section 6, we discuss how each node determines when to terminate neighbor discovery, without a priori knowledge about the number of its neighbors. We conclude in Section 7.

2. RELATED WORK

Neighbor discovery has been previously studied in the context of RF wireless networks; in particular, the focus has been on cases where directional communications are the norm. Most of the proposed approaches use random access as a basis for neighbor discovery. In [3], the authors examine the frequency with which a node should transmit neighbor discovery probes in order to maximize the probability of discovering a neighbor within a preset time period. They find that this probability is influenced by the density of nodes in the network (the number of neighbors of a node) and the beamwidth used for transmissions. A TDMA based neighbor discovery algorithm is proposed for 3-D RF wireless networks in [4]; the neighbor discovery is based on establishing a handshake between the neighbors that are trying to discover each other. However, the approach assumes that each node is equipped with a Global Positioning System (GPS) device. The authors suggest that nodes begin with using low powers so as to discover *close by* neighbors in interference reduced settings; the power is gradually increased to discover neighbors that are further away. Luo et al. analyze neighbor discovery in a CDMA-like system; they assume that each node is aware of its neighbors' identities and their spreading codes a priori. An energy efficient neighbor discovery protocol is proposed in [5]. Jakllari et al. propose an integrated neighbor discovery and MAC protocol for an ad hoc network with directional antennas [9]. In their work, they explicitly consider mobility unlike in the other efforts. Aloha-based neighbor discovery protocols for RF wireless networks are comprehensively analyzed in [2]. The authors propose a collision detection mechanism to improve performance. The algorithms proposed also allow nodes to determine when to terminate neighbor discovery, without any prior knowledge of their neighborhoods.

None of the above approaches can be directly applied in a ultraviolet wireless network due to its inherent physical (PHY) layer properties. Specifically, almost all of these protocols assume the existence of LOS links; they do not consider the possibility of transmitting in different directions to establish NLOS links (as is possible with UV) and thus, do not provide a relative ranking of these links in terms of the quality of communications. Second, they do not account for the inherent properties of the UV scattering channel (discussed later). Finally, many of them implicitly assume either prior knowledge of a node's neighborhood or specialized equipment such as GPS.

The first work (the only work to the best of our knowledge) on neighbor discovery in UV wireless networks is found in [7]. A credit-collection protocol is designed to rank the transmission directions of a node for each neighbor, based on packet losses on each directional link. This protocol requires a large number of packet transmissions to achieve stable results and it takes a relatively long time to terminate. Our goal here is to design a much faster protocol for neighbor discovery to obtain the same information about neighbors as in [7].

3. UV LINK AND TRANSCEIVER MODELS

In this section, we describe the propagation characteristics of UV communications. UV signal delivery is realized via the propagation of photons between the transmitter (made up of LEDs) and the receiver (referred to as the PMT as discussed earlier). The UV photons are scattered one or more times after being emitted by the LEDs. The atmospheric scattering is governed by a stochastic process that one has no control on, or exact knowledge of how many times the photons are scattered or in which direction they are scattered in the atmosphere. Note here that before the photons are captured by the PMT, their propagation direction might change multiple times due to what is defined as multiple scattering. Stated otherwise, a LOS link is not mandatory for UV communication although a LOS link can provide relatively lower path loss as has been verified in [10]. Although the path loss model we adopt in this paper considers multiple-scattering effects, the general propagation trends that we describe are consistent with single-scattering so that we can use geometric demonstrations to facilitate understanding. Figure 1 depicts a typical NLOS UV communication link. As long as there is a *common volume* between the transmission beam cone and the receiving field of view (FOV), a communication link can be potentially established.

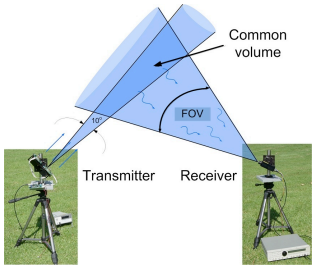

Figure 1: Common (overlap) volume between the transmission beam and the receiving field of view (FOV) enables a NLOS link.

Some path loss models have been proposed under the assumption that photons are scattered only once, which is *approximately* correct when the *optical length* is relatively short. The optical length is defined as the product of actual distance between the transmitter (Tx) and the receiver (Rx) and what is referred to as the scatter coefficient [10]. In our work, we adopt the path loss model which captures both single-scattering and multiple-scattering effects. More importantly, the model can capture arbitrary transmission directions from the LEDs. Thus the scenarios we study are not limited to the case where the transmission beam axis and the FOV axis are on the same plane (coplanar case). We introduce an additional angle to describe the noncoplanar case as shown in Fig. 2. The path loss model is developed in [8] by fitting a curve to experimentally obtained data on a real UV test-bed. A typical transceiver configuration is presented in Fig. 4. The receiver is designed to point vertically and the transmitter can orient itself any direction by turning on the corresponding LEDs on specific *facets*. The corresponding link loss model is adopted throughout this paper.

The empirical model is presented in mathematical form below:

$$L = \xi r^{\alpha} e^{b\varphi}. \quad (1)$$

The values of the path loss factor ξ, the path loss exponent α and

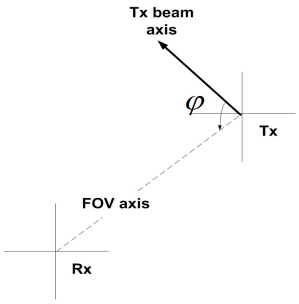

Figure 2: Transmission beam axis and FOV axis are located on the different planes. The off-axis angle φ is introduced to describe the offset between these two planes.

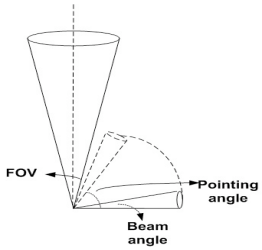

Figure 3: Beam angle, receiver FOV and pointing angle.

the off-axis angle exponent factor b are related to the transmission beam divergence, the FOV and the Tx/Rx pointing angles. Extensive experiments on a real UV Physical layer test-bed have been conducted to obtain these parameters. In our simulations below, we set these parameters to be those associated with the case where, the transmission pointing angle is equal to 10°. They are, $\xi = 5 \times 10^9$, $\alpha = 0.4$, $b = 8.7$, and $\varphi \in [0, \pi]$.

As is typical with UV transceivers, we assume that the Tx and the Rx are co-located on a communication node. Multiple transmitters are simultaneously activated to enable omni-directional transmissions. Each Tx consists of one or more LEDs, which are mounted on the side facets of a node as shown in Fig. 4. The LEDs installed on each facet transmit the same information-bearing signal when they are turned on. Transmitters are switched on or off digitally; no mechanical motion is needed to steer beams. The Rx is deployed on the top of the node to achieve omni-directional reception, i.e., the receiving pointing angle is 90° and Rx vertically faces upwards. A practical transceiver is not limited to the configuration in Fig. 4. The number of directions could be varied adaptively based on different application requirements and optical device settings.

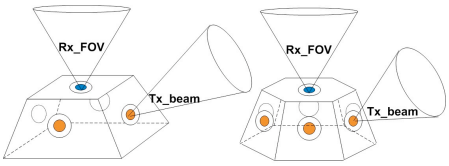

Figure 4: A UV transceiver with 4 (left) and 6 (right) transmission directions. Dots on the side facets are the directional LED transmitters; dots on the top facet are the omni-directional photon detectors.

With this design, the transceiver consists of multiple **directional transmitters** and an **omni-directional receiver**. The multiple transmitters can function individually to provide directional communication links. In other words, each target receiver can be associated with *one* best direction in terms of the lowest path loss. The transmitters can also work simultaneously to achieve omni-directional transmissions, which is useful when one needs to realize a broadcast function.

As previously discussed, atmospheric scattering cannot be controlled artificially. Some photons emitted by the Tx might be scattered back to its co-located Rx. This is defined as backscattering, which results in self-interference. Li et al. have conducted experiments to verify that as long as the Tx pointing angle is relatively small to make the beam depart from the FOV, the self-interference is negligible [8]. The basis for this conclusion is that in such scenarios, there is no (or very little) overlap between the Rx-FOV and the co-located Tx-beam as shown in Figs. 3 and 4. Since the transceiver is free of self-interference, spatial full-duplex communications are feasible with above the transceiver configuration. This feature greatly improves the throughput of UV communications and is a differentiating factor compared to RF communications (typically half-duplex).

4. OUR LEADERSHIP-BASED NEIGHBOR DISCOVERY PROTOCOL

In this section, we first elaborate the previously proposed neighbor discovery protocol for a UV network in [7]. We motivate and then, detail the design of our approach.

The first and the only neighbor discovery protocol (to our best knowledge) for a UV network is proposed in [7]. It makes no assumptions about a priori knowledge, such as the number of neighbors or the relative positions of neighbors. The basic procedure is described below:

- The Rx of every node is in a standby state all the time; it can receive anytime thanks to the full-duplex capability.
- A node randomly chooses a direction to send a packet after waiting for an exponentially distributed period.
- The transmitted packet could be either a request packet or a feedback packet. The request packet contains information about the request sender's ID and the ID of the direction that is being currently used. The feedback packet is composed of feedback sender's ID, request sender's ID, the IDs of the directions, and the total counts of received request packets in corresponding directions.
- When a node receives a request packet from a sender, it schedules a feedback packet to be returned to that sender. If no request packet from other nodes is received within a fixed pre-determined time period, the node will send out a request packet in a randomly chosen direction.
- Each node counts the number of received feedback packets and categorizes them into different groups based on the information carried in such packets. To elucidate, once request sender receives a feedback packet in return, it is aware of the recipient of the request, as well as the corresponding direction in which the request was received. The sender then adds a *credit* to that direction.
- A node ranks the directions for each of its neighbors, based on the number of credits accumulated for each direction. The direction with the highest number of credits is the best direction for that neighbor.

As may be evident from the above description, a node needs to collect a sufficiently large number of credits in order to accurately rank the directions towards a given neighbor. Since the primary

mode of communication is random access, collisions occur. This can skew the results since small sample sets may not yield accurate estimates of the best direction. In other words, this results in short term inaccuracies. Convergence to correct results takes a relatively long period of time.

Given this limitation, and the importance of expeditious neighbor discovery, we seek to design an interference-free environment for the process. Our goal here is to enable the discovery of neighbors and the best direction for communications with each neighbor with a *one-time packet exchange*. We later show that our method (proposed below) drastically improves the performance of neighbor discovery in terms of the time taken, compared to the method proposed in [7].

Our proposed approach: In a nutshell, our proposed approach relies on sequential neighbor discovery. A single node (the leader) initially performs neighbor discovery; after that node is done, the leadership is passed on to a second node and so no. We call our approach the "leadership-based neighbor discovery" protocol. In more detail, our approach consists of the following steps:

- Initially, all the nodes are in a standby state. They only receive (listen) packets but do not transmit anything.
- A pre-chosen leader node (could be randomly chosen) will then initiate the neighbor discovery process by sending a request packet in a randomly chosen direction. This request packet includes the ID of the sender and the direction in which it is transmitted.
- The leader waits for a fixed time period T, and then switches to the next direction to send the next request packet. It repeats the process until it performs packet transmissions in all possible directions.
- When a neighbor node receives a request packet from the leader node, it calculates the path loss experienced. Note that this is possible because there is no other concurrent transmission. (This is a key difference from the method in [7]; the path loss cannot be obtained since transmissions are subject to interference.)
- Each such neighbor waits for a period t; this period is randomly chosen in $[0, T]$. After this period, the neighbor randomly chooses a direction using which it transmits a feedback packet back to the leader. The feedback packet contains the ID of responding node and the path loss estimated using the request packet. The reception of the feedback packets is not guaranteed because contention exists among different neighbors that attempt sending feedback. However as long as the length of the feedback packets is much shorter than maximal waiting time T, it is likely to receive many of these feedback packets with a high probability.
- After the process is complete, the leader node can rank the directions for each discovered neighbor found based on the path loss information contained in feedback packets. It then randomly selects a successor from the nodes in its neighbor list. It sends a notifier packet to the selected successor node, using the best direction recorded for this node. The successor then performs neighbor discovery. All other nodes are made aware, that the neighbor discovery *token* has now been passed on to a different node.

It is possible that the notifier packet gets lost; to account for this, if no request packets are heard from the successor for a period $2T$, the former leader node will re-select a new successor. The process may have to be repeated in extreme cases until a successful leadership transfer is achieved. The steps described above are formalized with the pseudo code below:

```
K = total no. of directions;
Leader:
while rotation not finished yet do
  if time T is up then
    ND_request.source_ID=self_id;
    ND_request.direc_ID=uniformly_choose(K);
    send ND_request;
    set T;
  else
    if receive ND_feedback then
      neighbor_ID=ND_feedback.source_ID;
      prev_direc=ND_feedback.prev_direc;
      PathLoss=ND_feedback.path_loss;
      PathLoss.node[neighbor_ID].direc[prev_direc]=PathLoss;
    end if
  end if
end while
# of discovered neighbors is summarized here;
while i < # of discovered neighbors do
  order PathLoss.node[i].direc[*];
end while
select leader=uniformly_choose(# of discovered neighbors);
send Notifier packet to next leader;
Other nodes:
if receive ND_request then
  dest_ID=ND_request.source_ID;
  prev_direc=ND_request.direc_ID;
  PathLoss=Tx_power/Rx_power;
  set t=uniformaly_choose(T);
  if time t is up then
    ND_feedback.source_ID=self_id;
    ND_feedback.prev_direc=prev_direc;
    ND_feedback.path_loss=PathLoss;
    send ND_feedback;
  end if
end if
```

Evaluations of our approach: We evaluate the performance of our neighbor discovery protocol via extensive simulations performed using OPNET version 16.0 [12]. The simulation settings are listed in Table 1. The chosen transmission power corresponds to the UV transmission range (approximately 50m in Fig. 5). In all of the following simulations, we assume that the transceiver has 6 directions for transmission operations. The more the directions, the larger the neighbor table maintained and the longer the neighbor discovery process takes. The relation between the number of directions and the time consumed by the neighbor discovery process is discussed later. We adopt the channel model from Section 3 to characterize signal propagation. By default, the chosen parameters correspond to the scenario when the transmission pointing angle is equal to 10°. To recall, these parameters are: $\xi = 5 \times 10^9$, $\alpha = 0.4$, $b = 8.7$. The SIR threshold is set to be 1dB; the signal is detectable if the power is larger than the total interference power by this value. The sizes chosen for the three kinds of packets are consistent with that by IEEE 802.11 [11]; we choose this since protocols for UV networks are yet to be standardized. Each node is placed randomly in a network area of $1 \times 10^4 \text{m}^2$. The total number of nodes in the

network is 10. All the simulation curves are computed as averages of over 10 runs.

Table 1: Parameters settings in simulations

Transmission power	4 mW
Number of directions	6
Transmission pointing angle	10°
SIR threshold	1 dB
Collision model	Physical (accumulative) model
Traffic pattern	10000 pkt/sec
Request packet size	32 bits/pkt
Feedback packet size	80 bits/pkt
Notifier packet size	32 bits/pkt
Network size	100 m by 100 m
Number of nodes	10

We compare the newly proposed protocol (labeled leadership-based) with the previously proposed protocol from [7] (labeled credit-collection) in Fig. 5. We would like to point out here that this comparison is with the fastest version of the algorithm proposed in [7]. The expected value of the waiting interval λ for credit-collection is set to be 10^{-3}s. The uniformly distributed waiting interval T with the proposed protocol is set to be 2×10^{-3}s; this ensures that the mean is 10^{-3}s and thus, the settings are consistent in both cases. In Fig 5, The leadership-based algorithm leads to the sharp speed up in the neighbor discovery process. It saves about 92% of the time consumed in neighbor discovery, if one were to have a requirement that every node find 70% of its neighbors in the network. It saves about 97% of the time consumed in neighbor discovery if this requirement is stricter and every node has to find about 80% of its neighbors. The credit collection protocol consumes a lot of time to stabilize. These results suggest that interference has a dominant influence on neighbor discovery in UV networks; using a relatively interference-free approach drastically speeds up the neighbor discovery process.

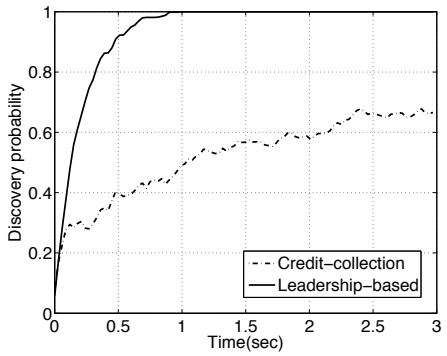

Figure 5: Comparison between leadership-based algorithm and credit-collection algorithm proposed in [7].

5. IMPROVING FAIRNESS

Although not explicitly stated earlier, we wish to point out here that a node can assume the role of the leader multiple times. We expect that the neighbor discovery process is a background process that is continuously executed; this is essential since the neighborhoods of nodes could change either due to failures, environmental changes (obstacles causing links to fail) or mobility. In this section, we consider the fairness of our approach in terms of the number of times a node gets an opportunity to become the leader.

To examine the fairness of our approach, we perform a simulation experiment, wherein we run the discovery process until all nodes discover almost all of their neighbors (> 99 %). The total simulation time was 3 seconds and node '0' was arbitrarily chosen to be the initial leader. Table 2 shows the number of times that each node is chosen to be the leader (referred to as *leader records*). We immediately see that certain nodes have a better chance in neighbor discovery; in other words, an imbalance is observed. Node 1 is chosen to be the leader 29 times; however, node 8 is only chosen 12 times. Table 3 shows the leader records up to the first 0.54 seconds of the simulation. In this period, the percentage of neighbors discovered is on average 95% and 7 of the nodes had found all of their neighbors. We observe that node 0 and node 4 assumed the role of the leader 6 times each; many other nodes had the opportunity only twice. We also indicate in this table, the percentage of neighbors discovered within this time. We observe that node 8 has discovered all of its neighbors even though it has been a leader just twice. Most nodes find all of their neighbors as long as they become a leader more than twice. This observation is encouraging in that, a node only has to serve as a leader a few times, in order for it to find its neighbors. If a node is chosen as a leader too often, it is likely to waste time and resources. Thus, it appears meaningful to provide equal opportunities of being a leader to all nodes in the network.

Table 2: Records of being the leader (time length = 3 secs)

Find 100% neighbors					
Node ID	0	1	2	3	4
Leader counts	22	29	14	13	18
Node ID	5	6	7	8	9
Leader counts	21	22	18	12	25

Table 3: Records of being the leader (time length = 0.54 secs)

Node ID	0	1	2	3	4
Leader counts	6	4	2	3	6
Neighbor percentage	1	1	0.75	0.89	1
Node ID	5	6	7	8	9
Leader counts	5	3	2	2	5
Neighbor percentage	1	1	0.81	1	1

A modified approach to improve fairness: In order for all nodes to equally share the leadership duties, intuitively it would seem that nodes need to have a *count* of how many times each has served as a leader. Then, the leadership role can be passed onto the neighbor who has served the least number of times. This will ensure that the number of times that each node serves as the leader will be roughly equivalent.

To achieve this goal, we incorporate what we call *a leader table* in the notifier packet. This enlarges the size of notifier packets compared to what was used earlier (as in Table 1). Specifically, the size of the packet will increase by $(\#of\ nodes \times 8\ bits)/packet$. The first leader node (also called trigger node), say node 0, will create this table and mark itself as being the leader once. Once the notifier packet is received by the next leader node (say node 1), a new entry is added into the table to record that node 1 served as the leader once. This leader table is updated at each leadership transfer instance. In this way, we ensure that the current leader node possesses the most up-to-date leader table. Each leader node will choose the next leader with the least *experience* of being the leader. If multiple nodes are qualified to be the next leader, one of them is

randomly chosen. The pseudo code below describes the process in detail.

```
K = total no. of directions;
Leader:
extract Leader_table from Notifier packet;
while rotation not finished yet do
  if time T is up then
    ND_request.source_ID=self_id;
    ND_request.direc_ID=uniformly_choose(K);
    send ND_request;
    set T;
  else
    if receive ND_feedback then
      neighbor_ID=ND_feedback.source_ID;
      prev_direc=ND_feedback.prev_direc;
      PathLoss=ND_feedback.path_loss;
      PathLoss.node[neighbor_ID].direc[prev_direc]=PathLoss;
    end if
  end if
end while
# of discovered neighbors is summarized here;
while i < # of discovered neighbors do
  order PathLoss.node[i].direc[*];
end while
select next_leader=Min(Leader_table[neighbor_ID]);
send Notifier packet to next_leader;
Other nodes:
if receive ND_request then
  neighbor_ID=ND_request.source_ID;
  prev_direc=ND_request.direc_ID;
  ND_feedback.path_loss=Tx_power/Rx_power;
  t=uniformly_choose(T);
  if t is up then
    ND_feedback.source_ID=self_id;
    ND_feedback.prev_direc=prev_direc;
    send ND_feedback;
  end if
end if
```

Evaluating the modified approach: Next, we present evaluations of our modified approach. Tables 4 and 5 show the number of times that each node in the network serves as a leader, when the overall percentage of neighbors discovered is approximately 100% and 95%, respectively. The results are from one randomly chosen realization. We observe that the variance in the counts (in Table 4) is relatively small; specifically, it is 0.44 compared to 26.84 in Table 2 with the original scheme. We see from Table 5 that all the nodes have been the leader, the same number of times; this means that the approach is fair. Both Table 3 and Table 5 reflect scenarios where each node found approximately 95% of its neighbors, successfully. However, the time incurred with the latter algorithm is only 0.42s; this represents a 22% reduction in time compared to the original approach. This reduction is a direct artifact of the increased efficiency with the modified scheme; nodes that have previously served as leaders and have found their neighbors do not waste time in serving as leaders again.

Figure 6 depicts how the modified scheme improves the performance of neighbor discovery. We choose a 20-node scenario and the area of deployment is still 100 m×100 m. The maximum transmission range and transmission power are 50 m and 0.4 mW, respectively; these are the same as in Table 1. From Fig. 6. We see that the time needed by every node for finding 95% its neighbors

Table 4: Records of being the leader (time length = 3 secs)

Find 100% neighbors					
Node ID	0	1	2	3	4
Leader counts	20	19	20	20	18
Node ID	5	6	7	8	9
Leader counts	19	19	19	20	20

Table 5: Records of being the leader (time length = 0.42 secs)

Node ID	0	1	2	3	4
Leader counts	3	3	3	3	3
Neighbor percentage	0.89	1	1	1	0.94
Node ID	5	6	7	8	9
Leader counts	3	3	3	3	3
Neighbor percentage	1	1	1	0.71	1

on average, with the basic leadership-based algorithm is about 1.62 seconds; this time is reduced to about 0.9 seconds with modified fair scheme. The modified scheme brings about a 45% performance improvement *after* taking into account the increase in packet transmission time due to the enlarging of the notifier packet.

Finally, we examine how the time for neighbor discovery varies as we vary the number of nodes in the network. Figure 7 shows how the time required to discover 95% of the neighbors changes (on average), as we vary the number of nodes in the network. We retain the coverage area to be 100 m×100 m for each simulation point represented in the figure. The notifier packet length is also set to be identical to that in the scenario with 20 nodes; we do this to isolate the impact of node density from the incurred overhead changes from the notifier packets. Figure 7 shows that the more the nodes in the network, the longer it takes for neighbor discovery as one might expect. More importantly, one can conclude that the time consumption is *proportional* to the number of nodes. The relationship is almost linear as seen in the figure.

As described earlier, our leadership-based algorithm attempts to construct an interference-free environment for the leader node to find its neighbors. However, with an increase in node numbers or network size, the time taken for neighbor discovery will increase. We further increase the node density (see Fig. 8) and we find that the time taken grows almost linearly versus the number of nodes. For very dense networks, as compared with the credit-collection algorithm, we found that the savings in terms of time get larger with increased node density. We use a curve fitting method to obtain extrapolated results for the leadership-based (solid line) and the credit-collection (dotted line) protocols, respectively. The curve fitting expression for the solid line is $0.4264 \times \exp(0.02799n)$ and the expression for the dotted line is $0.4937 \times \exp(0.03880n)$, where n stands for the number of nodes in the network. The slopes for both curves after differentiation of these expressions are obtained as $0.4264 \times 0.02799 \times \exp(0.02799n)$ and $0.4937 \times 0.03880 \times \exp(0.03880n)$, respectively. For a fixed value of n, the slope of the curve for the credit-collection protocol is always larger than the one for the newly proposed leadership-based protocols. Thus, one might expect the difference between two curves to grow when the number of nodes increases; this can be observed in Fig. 8.

In order to cover all the directions seamlessly, the number of transmitters installed should be inversely proportional to the transmis-

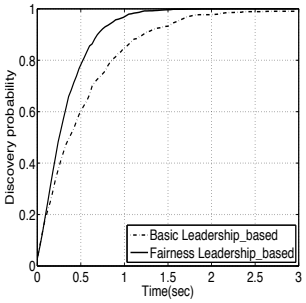

Figure 6: Comparison between schemes with or without fairness consideration.

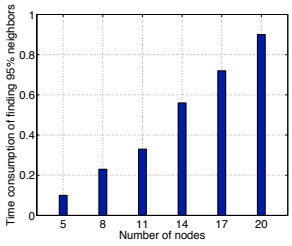

Figure 7: The time taken for neighbor discovery varies with number of nodes. The time shown is when 95% of neighbors are discovered.

sion beam divergence of each Tx. For example, if the beam divergence is 60^o, the number of transmitters should at least be 6 to cover all the directions. With more number of transmitters installed, the transmission beam divergence of each Tx can be reduced. Assuming that the transmission power is kept the same, small beam divergence results in longer transmission range and lower interference to other nodes. However, a large number of transmitters also implies that the time necessary for the neighbor discovery increases because the nodes have to search and rank more directions. Figure 9 shows the relationship between the time required and the number of transmitters. Each direction corresponds to one transmitter individually; for e.g. the plot corresponding to 15 directions represents the scenario where the transceiver is equipped with 15 facets and 15 transmitters. From this figure, we conclude that the more directions, the longer the neighbor discovery process. For the four cases compared in Fig. 9, in order to find 95% of neighbors on average, it takes 0.9 secs, 1.4 secs, 1.92 secs and 2.39 secs, for the cases with 6 directions, 9 directions, 12 directions, and 15 directions, respectively.

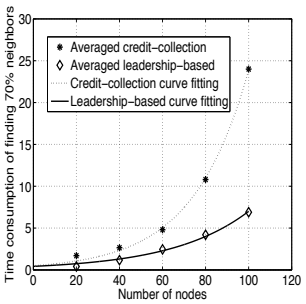

Figure 8: The time taken for neighbor discovery varies with number of nodes – a comparison of the leadership-based algorithm and the credit-collection algorithm. The time shown is when 70% of the neighbors are discovered, on average.

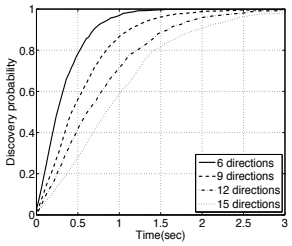

Figure 9: Performance with different numbers of directions. 20 nodes in total are distributed in the network.

6. TERMINATING NEIGHBOR DISCOVERY

In most application scenarios, it is important to bootstrap the network within a relatively short period. It is unlikely that a deployer will be able to pre-configure nodes with the specific topology; moreover, link qualities are likely to vary temporally. Our goal here is to determine when a neighbor discovery process can be considered to have terminated.

The scheme for terminating neighbor discovery: Achieving the above goal is far from trivial. Due to multi-user interference, feedback packets may collide, leading to information loss. As a consequence, a leader node is unlikely to realize all its neighborhood information by simply serving as the leader once. Thus, as discussed earlier, a node assumes the role of the leader multiple times. At each instance, it updates its neighborhood information with any data that is newly collected. If a node is unable to obtain new information upon becoming the leader, it is likely that it has already obtained its complete neighborhood information. In fact, the more times this happens, the more confidence one has in this conclusion. Thus, we make a modified rule that if a node does not acquire new information for a certain consecutive number of instances when it serves as a leader, it assumes that it does not need to be a leader anymore. It *marks* itself as being done with neighbor discovery. If all nodes have marked themselves as being done, then the procedure terminates. The information on which of the nodes are marked as above is recorded in a termination table. This table is exchanged in the notifier packets; thus, it is globally disseminated. We discuss this table later.

Each node maintains a counter which is initialized to '0'. Whenever it has acted as a leader but has not obtained any new neighborhood information, it increments this counter. If it does obtain new information, the counter is reset. When the counter value reaches a pre-defined threshold N, the node marks itself. When it transfers leadership, it updates the termination table to indicate that it is done with neighbor discovery. A simple bit corresponding to the nodes' IDs can be set to indicate that it is done. In other words, the termination table can simply be a bitmap which is equal in size to the number of nodes in the network.

With the information in the termination table, the current leader has the most up-to-date information and can make a timely and efficient decision on which node to choose as the leader next. Note here that this can be combined with the fairness criterion described earlier; however, we use this as the explicit criteria in the remainder of the section.

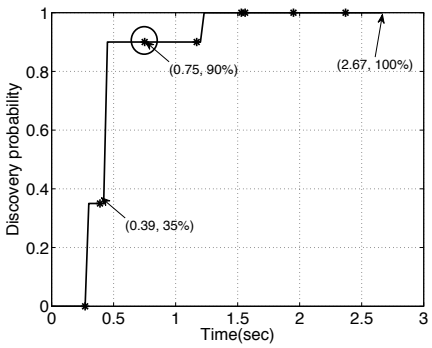

Figure 10: Illustrating the termination process. Solid line shows how the percentage of discovered neighbors increases with time; '*'s represent the instances when the node starts serving as the leader.

Evaluations: Figure 10 shows the percentage of neighbors discovered with time for a single realization from the perspective of an

arbitrarily chosen node in the network. We simulate the 20-node scenario and other simulation settings are the same as described in Section 5. In this figure, we see that the node under consideration found 35% of its neighbors after the first instance of being the leader; this percentage increased to 75% after its second instance of being the leader. However, as shown by the circle in the plot, after the node served as the leader for the third time, there was no new information obtained; note however that, the neighbor discovery was not complete for the node. Although the counter increased by 1, it was reset after the node served as a leader for the fourth time, since new information was acquired. The neighbor discovery procedure finally ended at 2.67 seconds because the counter threshold was reached. At that point, the node had indeed discovered all of its neighbors.

Table 6: Leader records after finding 100% neighbors

Node ID	0	1	2	3	4
Leader counts	6	4	5	7	4
Node ID	5	6	7	8	9
Leader counts	5	4	5	6	6
Node ID	10	11	12	13	14
Leader counts	5	6	6	4	2
Node ID	15	16	17	18	19
Leader counts	7	7	7	6	7

Table 6 shows the total leader counts *after* every node found 100% of its neighbors. We set the termination counter threshold mentioned above to be 2. Interestingly, from the Table 6 we observe that all the nodes except the one with node ID of 14 served as the leader without any information update, more than 2 times. This is because, node 14 was the final node in the neighbor discovery process; unless its counter was set to 2, the process could not be terminated. However, node 14 could be assigned to being a leader only by one of its neighbors. In this case, nodes attempted to search for 14 and interim also served as leaders again (the leadership token was passed around). Finally, the token was passed onto node 14 and the process terminated.

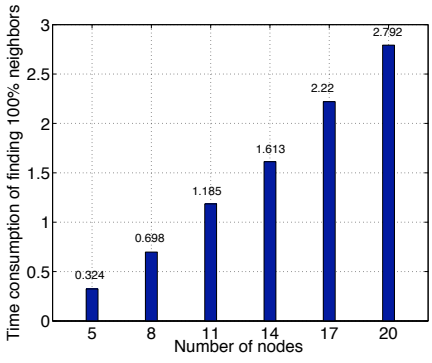

Figure 11: Illustration of the performance of the termination process with different node densities. All the nodes in the network find 100% of their neighbors.

Figure 11 shows the performance of the protocol with the termination procedure with six scenarios, with different node densities. We see the termination time is proportional to the node density as one may expect.

7. CONCLUSIONS

Neighbor discovery in UV networks is an important function that allows nodes to bootstrap the topology and begin operations. While there have been a plethora of proposals for neighbor discovery in the RF context, they are inapplicable in the UV context given its unique PHY characteristics. In this paper, we propose a novel neighbor discovery protocol for UV networks. The key idea is to eliminate interference by arbitrating the discovery process by using leaders; a leader is the only node that is allowed to perform neighbor discovery at any given time. The leadership role is passed on from node to node. We demonstrate that our protocol is extremely effective in performing neighbor discovery via extensive simulations that incorporate the UV PHY layer. In particular, the time needed for neighbor discovery process is decreased by as much as 90% as compared to the only previously proposed neighbor discovery scheme for UV networks. We also provide extensive evaluations to capture the impact of various network and system parameters on the performance of our protocol.

8. REFERENCES

[1] Z. Xu and B. M. Sadler. Ultraviolet communications: potential and state-of-the-art. *IEEE Commun. Magazine*, 46(5), May 2008.

[2] S. Vasudevan, D. Towsley, R. Khalili, and D. Goeckel. Neighbor discovery in wireless networks and the coupon collector's problem. In *ACM Mobicom*, 2009.

[3] S. Vasudevan, J. Kurose, and D. Towsley. On neighbor discovery in wireless networks with directional antennas. In *IEEE INFOCOM*, March 2005.

[4] G. Pei, M. Albuquerque, J. Kim, D. Nast, and P. Norris. A neighbor discovery protocol for directional antenna networks. In *IEEE MILCOM*, October 2005.

[5] M. J. McGlynn and S. A. Borbash. Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks. In *Mobihoc*, September 2001.

[6] J. Luo and D. Guo. Neighbor discovery in wireless ad hoc networks based on group testing. In *46th Annual Alerton Conf.*, September 2008.

[7] Y. Li, L. Wang, Z. Xu, and S. V. Krishnamurthy. Neighbor discovery for ultraviolet ad hoc networks. In *IEEE JSAC*, 2011 (accepted with revision).

[8] Y. Li, J. Ning, Z. Xu, S. V. Krishnamurthy, and G. Chen. Uvoc-mac: A MAC protocol for outdoor ultraviolet networks. In *IEEE ICNP*, October 2010.

[9] G. Jakllari, W. Luo, and S. Krishnamurthy. An integrated neighbor discovery and MAC protocol for ad hoc networks using directional antennas. *IEEE Trans. on Wireless Communications*, 6(3), 2007.

[10] G. Chen, Z. Xu, H. Ding, and B. M. Sadler. Path loss modeling and performance trade-off study for short-range none-line-of-sight ultraviolet communication. *Optics Express*, 17(5), March 2009.

[11] Wireless LAN medium access Control (MAC) and physical layer (PHY) specifications. http://standards.ieee.org/getieee802/download/802.11-2007.pdf.

[12] Opnet User's Documentation. http://www.opnet.com.