Neighbor discovery for wireless networks via compressed sensing☆

Lei Zhang, Jun Luo [1], Dongning Guo *

Department of Electrical Engineering & Computer Science, Northwestern University, 2145 Sheridan Road, Evanston, IL 60208, USA

ARTICLE INFO

ABSTRACT

This paper studies the problem of neighbor discovery in wireless networks, namely, each node wishes to discover and identify the network interface addresses (NIAs) of those nodes within a single hop. A novel paradigm, called *compressed neighbor discovery* is proposed, which enables all nodes to simultaneously discover their respective neighborhoods with *a single frame* of transmission, which is typically of a few thousand symbol epochs. The key technique is to assign each node a unique *on–off signature* and let all nodes simultaneously transmit their signatures. Despite that the radios are half-duplex, each node observes a superposition of its neighbors' signatures (partially) through its own off-slots. To identify its neighbors out of a large network address space, each node solves a *compressed sensing* (or *sparse recovery*) problem.

Two practical schemes are studied. The first employs random on–off signatures, and each node discovers its neighbors using a noncoherent detection algorithm based on group testing. The second scheme uses on–off signatures based on a deterministic second-order Reed–Muller code, and applies a chirp decoding algorithm. The second scheme needs much lower signal-to-noise ratio (SNR) to achieve the same error performance. The complexity of the chirp decoding algorithm is sub-linear, so that it is in principle scalable to networks with billions of nodes with 48-bit IEEE 802.11 MAC addresses. The compressed neighbor discovery schemes are much more efficient than conventional random-access discovery, where nodes have to retransmit over many frames with random delays to be successfully discovered.

1. Introduction

In many wireless networks, each node has direct radio link to only a small number of other nodes, called its *neighbors (or peers)*. Before efficient routing or other network-level activities are possible, nodes have to discover and identify the network interface addresses (NIAs) of their neighbors. This is called *neighbor discovery (or peer discovery)*. The problem is crucial in mobile ad hoc networks (MANETs), which are self-organizing networks without pre-existing infrastructure. The problem is becoming important in increasingly heterogeneous cellular networks with the deployment of unsupervised picocells and femtocells.

A node interested in its neighborhood, which is henceforth referred to as the *query node*, listens to the wireless channel during the discovery period, and then decodes the NIAs of its neighbors. Neighbors transmit signals which contain their identity information. It is fair to assume that non-neighbors either do not transmit, or their signals are weak enough to be regarded as noise. We make two important observations: (1) the physical channel is a multiaccess channel, where the

observation made by the query node is a (linear) superposition of transmissions from its neighbors, corrupted by noise; (2) the goal of neighbor discovery is to identify, out of all valid NIAs, which ones are used by its neighbors.

State-of-the-art neighbor discovery protocols, such as that of the IETF MANET working group [1] and the ad hoc mode of IEEE 802.11 standards, can be described as follows. The query node broadcasts a probe request. Its neighbors then reply with probe response frames containing their respective NIAs. If a response frame does not collide with any other frame, the corresponding NIA is correctly received. Due to lack of coordination, each neighbor has to retransmit its NIA enough times with random delays, so that it can be successfully received by the query node with high probability despite collisions. We refer to such a scheme as *random-access neighbor discovery*. Several such algorithms which operate in or on top of medium access control (MAC) layer have been proposed [2–7].

Random access assumes a specific signaling format, namely, a node's response over the discovery period basically consists of repetitions of its NIA interleaved with periods of silence. This signaling format allows the NIA to be directly read out from a successfully received frame. Every node can discover its neighborhood and also be discovered by neighbors given long-enough discovery period. However, such signaling is far from optimal. To design the optimal signaling, we should remove all unnecessary structural restrictions on the responses. Given the duration of the discovery period, the problem is in general to assign each node a distinct response, or *signature* over that period, and to design a decoding algorithm for a query node to identify the constituent signatures (or corresponding NIAs) based on the observed superposition. It would be ideal if all the signatures were orthogonal to each other, but this is impossible in case the number of signatures far exceeds the signature length. A good design should make the correlation between any pair of signatures as small as possible.

A crucial observation is that the number of actual neighbors is typically orders of magnitude smaller than the node population, or more precisely, the size of the NIA space, so that neighbor discovery is by nature a *compressed sensing (or sparse recovery)* problem [8,9]. By the wisdom from the compressed sensing literature, the required number of measurements (the signature length) is dramatically smaller than the size of the NIA space.

Based on the preceding observations, this work provides a novel solution, referred to as *compressed neighbor discovery*, which attains highly desirable trade-off between reliability and the length of the discovery period, thus minimizing the neighbor discovery overhead in wireless networks. The defining feature is to let nodes simultaneously transmit their signatures within a single frame interval. In order to let each node discover its own neighborhood during the same frame interval it is transmitting, i.e., to achieve full-duplex neighbor discovery, the signatures consist of on- and off-slots, so that within the discovery frame a node can make observations during its off-slots and also transmit during its on-slots. Some sparse recovery algorithm is then carried out to decode the neighborhood.

The organization of the remaining sections of the paper and our key contributions are as follows. After the system model is presented in Section 2, two types of signatures with corresponding decoding algorithms are proposed. The first scheme, which is studied in Section 3, assigns each node a pseudo-random *on–off signature* (i.e., a sequence of delayed pulses) over the (slotted) discovery frame. The number of on-slots is a small fraction of the total number of slots, so that the signature is sparse. The superposition of the signatures of all neighbors is a denser sequence of pulses, in which a pulse is seen at a slot if at least one of the neighbors sent a pulse during the slot. A simple decoding procedure via eliminating non-neighbors is developed based on algorithms originally introduced for *group testing* [10,11]. The complexity of the algorithm is linear in the size of the address space, which is feasible for networks with moderately large but not too large NIA spaces.

The second scheme, which is studied in Section 4, generates a set of deterministic signatures based on a second-order Reed–Muller (RM) code. First- and second-order RM codes date back to 1950s and are fundamental in the study of error-control codes and algorithms [12]. More recently, RM codes have been shown to be excellent for sparse recovery [13]. The original RM code consists of quadrature phase-shift keying (QPSK) symbols, with no off-slots. In order to achieve full-duplex discovery, we introduce off-slots by replacing roughly a half of the QPSK symbols by zeros. The chirp decoding algorithm of [13] is modified to perform despite the erasures. The choice of modified RM codes for neighbor discovery is not incidental: the algebraic structure allows unusually low decoding complexity (sublinear in the number of codewords), so that the scheme is in principle scalable to 2^{48} or more nodes or NIAs in the network [14].

In Section 5, compressed neighbor discovery is compared with random-access schemes and shown to require much fewer transmissions to achieve the same error performance. In addition, the new scheme entails much less transmission overhead (such as preambles and parity checks), because it takes a single frame of transmission, as opposed to many frame transmissions in random access.

We highlight some of the unique contributions of this work:

- It is the first to propose on–off signaling for achieving *full-duplex* neighbor discovery using half-duplex radios, which departs from conventional schemes where the transmitting frames of a node are scheduled away from its own receiving frames;
- This work is the first to use Reed–Muller codes for neighbor discovery, which enables highly efficient discovery for networks of any practical size;
- Previous work [15,16] only models the neighborhood of a single query node. This paper considers a more realistic network modeled by a Poisson point process, and a more realistic propagation model with path loss;
- The decoding algorithm for random on–off signatures significantly improves the performance of the group-testing-based algorithms studied in [15] and [16] for noiseless and Rayleigh fading channels, respectively;

- Previous work [16] only demonstrates reliable discovery at high signal-to-noise ratio (SNR) for a rather sparse network in which the average number of neighbors is less than ten. Numerical results in this paper demonstrate reliable and efficient discovery of 30 neighbors or more at fairly low SNR.

2. The channel and network models

2.1. The linear channel

Consider a wireless network where each node is assigned a unique network interface address. Let the address space be $\{0, 1, \ldots, N\}$ (e.g., $N = 2^{48} - 1$ if the space consists of all IEEE 802.11 MAC addresses). The actual number of nodes present in the network can be much smaller than N, but as far as neighbor discovery is concerned, we shall assume that there are exactly $N + 1$ nodes.

We will later discuss the problem of having all nodes simultaneously discover their respective neighborhoods, but for now let us assume that node 0 is the only query node and sends a probe signal to prompt a neighbor discovery period of M symbol intervals. Each node n in the neighborhood responds by sending a signal $\boldsymbol{S}_n = [S_{1n}, \ldots, S_{Mn}]^\top$. The signal identifies node n and is also referred to as the *signature* of node n. In case a node only transmits over selected time instances, those symbols S_{mn} corresponding to non-transmissions are regarded as zeros. For the time being let us ignore the variation of the small propagation delays between the query node and its neighbors, and assume symbol-synchronous transmissions from all nodes. We also assume that this discovery period is shorter than the channel coherence time. The received signal of node 0 can thus be expressed as

$$\boldsymbol{Y} = \sqrt{\gamma} \sum_{n \in \boldsymbol{N}_0} U_n \boldsymbol{S}_n + \boldsymbol{W} \tag{1}$$

where $\boldsymbol{N}_0$ denotes the set of NIAs in the neighborhood of node 0, U_n denotes the complex-valued coefficient of the wireless link from node n to node 0, γ denotes the average channel gain in the SNR, and $\boldsymbol{W}$ consists of M independent unit circularly symmetric complex Gaussian random variables, with each entry $W_m \sim \mathcal{CN}(0, 1)$. For simplicity, transmissions from non-neighbors, if any, are accounted for as part of the additive Gaussian noise.

The goal is to recover the set $\boldsymbol{N}_0$, given the observation $\boldsymbol{Y}$, the SNR γ, and knowledge of the signatures $\boldsymbol{S}_1, \ldots, \boldsymbol{S}_N$. The random coefficients U_n are unknown except for its statistics. For convenience, we introduce binary variables B_n, which is set to 1 if node n is a neighbor of node 0, and set to 0 otherwise. Let $\boldsymbol{X} = [B_1 U_1, \ldots, B_N U_N]^\top$ and $\underline{\boldsymbol{S}} = [\boldsymbol{S}_1, \ldots, \boldsymbol{S}_N]$. Then model (1) can be rewritten as

$$\boldsymbol{Y} = \sqrt{\gamma} \underline{\boldsymbol{S}} \boldsymbol{X} + \boldsymbol{W} \tag{2}$$

where we wish to determine which entries of $\boldsymbol{X}$ are nonzero, i.e., to recover the support of $\boldsymbol{X}$.

Model (2) represents a familiar noisy linear measurement system. We shall refer to $\boldsymbol{Y} = [Y_1, \ldots, Y_M]^\top$ as the measurements, and $\underline{\boldsymbol{S}}_{M \times N}$ as the known signature matrix. It is reasonable to assume that $B_1, \ldots, B_N$ are independent and identically distributed (i.i.d.) Bernoulli random variables with $\mathsf{P}\{B_1 = 1\} = c/N$, where c denotes the average number of neighbors of node 0. Let us further assume that $U_1, \ldots, U_N$ are i.i.d. with known distribution, and are independent of $B_1, \ldots, B_N$ and noise. To recover the support of $\boldsymbol{X}$ is then a well-defined, familiar statistical inference problem.

The node population $N + 1$ is typically much larger than the number of symbol epochs in one discovery period M, so that the linear system (2) is under-determined even in the absence of noise. An important observation is that the vector variable $\boldsymbol{X}$ is very sparse, so that neighbor discovery is fundamentally a sparse recovery problem, which implies that very few measurements, which can be orders of magnitude smaller than N, are sufficient for reconstructing the N-vector $\boldsymbol{X}$ or its support [17].

2.2. Signatures and their distribution

In the case of random-access neighbor discovery, each $\boldsymbol{S}_n$ consists of repetitions of the NIA of node n interleaved with random delays, sufficient synchronization flags, training symbols and parity check bits are embedded so that the delays can be measured accurately (this constitutes substantial overhead).

In general, the signature of node n can be regarded as the n-th codeword from the codebook $\underline{\boldsymbol{S}}$. (In the case of delay uncertainty, we can use a larger codebook to include shifted versions of the signatures.) The signatures of all nodes, i.e., the codebook, should be known input to the neighbor discovery algorithm carried out by any query node.

To make the distribution of a large codebook to all nodes practical, some simple structure shall always be introduced. For example, the signature of node n can be generated using a common pseudo-random number generator with the seed equal to n. It then suffices to distribute the generator (e.g., as a built-in software/hardware function) in lieu of the signatures. In principle, each node can construct the codebook $\underline{\boldsymbol{S}}$ by enumerating all valid NIAs, so that all signatures are known to all nodes in advance without any communication overhead. It is also possible to design an inverse mapping to recover the index n given any signature $\boldsymbol{S}_n$. An alternative design is to let the signatures be codewords of an error-control code, in which case it suffices to reveal the code to all nodes.

A key finding of this paper is that, in order for efficient neighbor discovery, the signatures should not merely consist of repetitions of the NIA. Discovery using cleverly designed signatures is not only feasible, but can be significantly more efficient than random-access discovery.

2.3. Propagation delay and synchronicity

In general, a receiver has to resolve the timing uncertainty of its neighbors in order to recover their identities. By including sufficient synchronization flags, random-access schemes are robust with respect to arbitrary delays. Since it is costly to add enough redundancy to allow accurate estimation of the delays in a multiuser environment, it can be beneficial to let nodes transmit their signatures simultaneously and synchronously. Some common clock, such as access to the global positioning system (GPS) can provide the timing needed. In our scheme, it suffices to have all communicating peers be approximately symbol-synchronized, as long as the timing difference (including the propagation delay) is much smaller than the symbol interval. This can be achieved by using distributed algorithms for reaching average consensus [18].

By definition neighbors should be physically close to the query node, so that the radio propagation delay is much smaller compared to a symbol epoch. For instance, if neighbors are within 300 m, the propagation delay is at most 1 μs, which is much smaller than the bit or pulse interval of a typical MANET. More pronounced propagation delays can also be explicitly addressed in the physical model, but this is out of the scope of this paper.

Admittedly, synchronizing nodes requires an upfront cost in the operation of a wireless network. The benefit, however, is not limited to the ease of neighbor discovery, but includes improved efficiency in many other network functions. Whether synchronizing the nodes is worthwhile is a challenging question, which is not discussed further in this paper.

2.4. Propagation loss and near–far problem

In previous work [16], we considered a single query node and neighbors of the same distance, and simply assumed the channel gains U_n to be Rayleigh fading random variables. In this paper, we incorporate the effect of network topology and propagation loss in the channel model. Suppose all nodes transmit at the same power, large-scale attenuation follows power law with path loss exponent α, and small-scale attenuation follows i.i.d. fading. Due to reciprocity, the gains of the two directional links between any pair of nodes are identical.

From the viewpoint of a query node, it suffices to describe the statistics of U_n of neighboring nodes in model (1) as follows. Suppose all nodes are distributed in a plane according to a homogeneous Poisson point process with intensity λ. Consider a uniformly and randomly selected pair of nodes. The channel power gain between them is $GR^{-\alpha}$, where G denotes small-scale fading and R stands for the distance between them. The nodes are called neighbors of each other if the channel gain between them exceeds a certain threshold, i.e., $GR^{-\alpha} > \eta$ for some fixed threshold η. We choose not to define the neighborhood purely based on the geometrical closeness because (1) connectivity between a pair of nodes is determined by the channel gain; and (2) a receiver cannot separate the attenuations due to path loss and Rayleigh fading in one discovery period.

Consider an arbitrary neighbor, n, of the query node, where the distance between them is R, and the random attenuation of the channel is G. By definition of a neighbor, G and R must satisfy $GR^{-\alpha} \geq \eta$, i.e., $R \leq (G/\eta)^{1/\alpha}$. Under the assumption that all nodes form a Poisson point process, for given G, this arbitrary neighbor n is uniformly distributed in a disc centered at the query node with radius $(G/\eta)^{1/\alpha}$. Therefore, the conditional distribution of R given G can be expressed as

$$P(R \leq r | G) = \begin{cases} r^2 \left(\dfrac{\eta}{G}\right)^{\frac{2}{\alpha}}, & r \leq \left(\dfrac{G}{\eta}\right)^{\frac{1}{\alpha}}; \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Now for every $u \geq \sqrt{\eta}$, by (3) we have

$$P(GR^{-\alpha} \geq u^2) = \mathsf{E}_G \left\{ P\left(R \leq \left(\dfrac{G}{u^2}\right)^{\frac{1}{\alpha}} \middle| G \right) \right\}$$

$$= \mathsf{E}_G \left\{ \left(\dfrac{\eta}{u^2}\right)^{\frac{2}{\alpha}} \right\}$$

$$= \dfrac{\eta^{\frac{2}{\alpha}}}{u^{\frac{4}{\alpha}}}. \tag{4}$$

Hence the probability density function (pdf) of $|U_n|$ of neighbor n is

$$p(u) = \begin{cases} \dfrac{4}{\alpha} \dfrac{\eta^{2/\alpha}}{u^{4/\alpha+1}}, & u \geq \sqrt{\eta}; \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Interestingly, the distribution does not depend on the fading statistics (of G). Moreover, it is fair to assume that the coefficients are circularly symmetric, i.e., the phase of U_n is uniform on $[0, 2\pi)$.

Without loss of generality, we assume that the query node locates at the origin. Denote $\Phi = \{X_i\}_i$ as the point process consisting of all nodes excluding the query node. By Slivnyak–Meche's theorem [19], Φ is a Poisson point process with intensity λ. Fading coefficients can be regarded as independent marks of the point process, so that $\tilde{\Phi} = \{(X_i, G_i)\}_i$ is an independently marked Poisson point process. By Campbell's theorem [19], the average number of the query node can be obtained as

$$
\begin{aligned}
c &= \mathsf{E}_{\tilde{\Phi}} \left\{ \sum_{(X_i, G_i) \in \tilde{\Phi}} \mathbf{1} \left(G_i R_i^{-\alpha} \geq \eta \right) \right\} \\
&= 2\pi\lambda \int_0^\infty \int_0^\infty \mathbf{1} \left(G R^{-\alpha} \geq \eta \right) R e^{-G} \mathrm{d}R \mathrm{d}G \\
&= \frac{2}{\alpha} \pi \lambda \eta^{-2/\alpha} \Gamma \left(\frac{2}{\alpha} \right)
\end{aligned}
\tag{6}
$$

where $\mathbf{1}(\cdot)$ is the indicator function and $\Gamma(\cdot)$ is the Gamma function.

The near–far situation, namely that some neighbors can be much stronger than others is inherently modeled in (1)–(5). The proposed sparse recovery algorithms are highly resilient to the near–far problem. In particular, in the case of deterministic signatures, the gain of strong neighbors can be estimated quite accurately so that their interference to weaker neighbors can be removed.

2.5. Network-wide discovery

Unlike in previous work [15,16], this paper also considers the problem that many or all nodes in the network need to discover their respective neighborhoods at the same time. A major challenge is posed by the *half-duplex* constraint, i.e., a wireless node cannot receive any useful signal at the same time and over the same frequency band on which it is transmitting [20,21]. This is due to the limited dynamic range of affordable radio frequency circuits. Standard designs of wireless networks use time- or frequency-division duplex to schedule transmissions of a node away from the time-frequency slots the node employs for reception [22].

A random-access scheme naturally supports network-wide discovery. This is because each node transmits its NIA intermittently, so that it can listen to the channel to collect neighbors' NIAs during its own epochs of non-transmission. Collision is inevitable, but if each node repeats its NIA a sufficient number of times with enough (random) spacing, then with high probability it can be received by every neighbor once without collision.

As we shall see in Sections 3 and 4, the proposed compressed neighbor discovery schemes employ on–off signatures, so that a node can make observations during its own off-slots. All nodes broadcast their signatures and discover their respective neighbors at the same time. Thus network-wide discovery is achieved within a single frame interval.

3. Random signatures and group testing

In this section, we consider using random on–off signatures. Specifically, the measurement matrix $\underline{S}$ consists of i.i.d. Bernoulli random variables, with $\mathsf{P}(S_{mn} = 1) = 1 - \mathsf{P}(S_{mn} = 0) = q$ for all m, n. As aforementioned, the signatures can be generated using a common pseudo-random number generator so that $\underline{S}$ is known to all query nodes.

3.1. A previous algorithm based on group testing

In the absence of noise, neighbor discovery with on–off signatures is equivalent to the classical problem of group testing. In fact, group testing has been used to solve a related RFID problem [23] and a multiple access problem [11,24]. For every $m = 1, \ldots, M$, the measurement $Y_m = \sqrt{\gamma} \sum_{n=1}^N S_{mn} X_n$ is nonzero if any node from the group $\{n : n = 1, \ldots, N,$ and $S_{mn} \neq 0\}$ is a neighbor. Algorithm 1 visits every measurement Y_m with its power below a threshold T to eliminate all nodes who would have transmitted energy at time m from the neighbor list. Those nodes which survive the elimination process are regarded as neighbors.

Algorithm 1 Simple group testing

1: **Input:** $\boldsymbol{Y}$, $\underline{\boldsymbol{S}}$ and T
2: **Initialize:** $V \leftarrow \{1, \ldots, N\}$
3: **for** $m = 1$ to M **do**
4:    **if** $|Y_m|^2 < T$ **then**
5:       $V \leftarrow V \setminus \{n : S_{mn} = 1\}$
6:    **end if**
7: **end for**
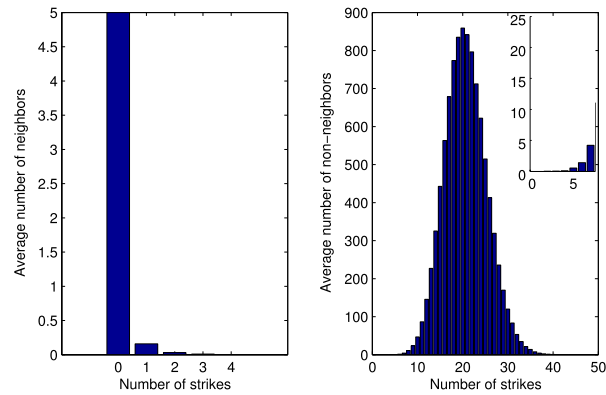8: **Output:** mark all nodes in V as neighbors

---

Fig. 1. Histograms: The average number of nodes versus the number of strikes they receive as a neighbor (the left plot) or non-neighbor (the right plot). The inlet plot amplifies the left-side tail. SNR = 20 dB, $c = 5$.

Two types of errors are possible: If an actual neighbor is eliminated by the algorithm, it is called a *miss*. On the other hand, if a non-neighbor survives the algorithm and is thus declared a neighbor, it is called a *false alarm*. The *rate of miss* (resp. *rate of false alarm*) is defined as the average number of misses (resp. false alarms) in one node's neighborhood divided by the average number of neighbors a node has.

Algorithm 1 requires only noncoherent energy detection and is remarkably simple. However, discovery is reliable only if the SNR is high and the average number of neighbors a node has is very small, whereas the error performance is unacceptable for many practical scenarios [16].

3.2. Improvements: t-tolerance test and phase randomization

The improved scheme proposed in this section is based on Algorithm 1 and includes two major changes. First, instead of eliminating a node as soon as it disagrees with one measurement, multiple disagreements are needed to eliminate a node. Second, to decouple the measurements, we randomize the phase of the samples of each signature.

It is instructive to examine the events which trigger elimination. To this end, we record, for each node eliminated, the number of near-zero measurements which point to its elimination, which are referred to as *strikes*. Fig. 1 illustrates the average number of nodes (out of 10,000 total) which receive 0, 1, 2, . . . strikes as neighbors or non-neighbors, respectively. It turns out that most of the time a neighbor agrees with all measurements and hence receives no strike, but occasionally a neighbor may receive 1, 2 or 3 strikes due to noise or mutual cancellation. In contrast, most non-neighbors receive dozens of strikes because they disagree with many measurements, whereas a small number of non-neighbors receive fewer than 5 strikes. Algorithm 2, which is referred to as the t-tolerance test [25], allows a node to survive up to t strikes, and requires that a node be eliminated only if it receives strictly more than t strikes. By tuning the number t, one can select the most desirable trade-off between the rate of miss and the rate of false alarm.

Algorithm 2 t-tolerance group testing

1: **Input:** Y, $\underline{S}$ and T
2: **Initialize:** $v_n \leftarrow t + 1, n = 1, \ldots, N$
3: **for** $m = 1$ to M **do**
4:    **if** $|Y_m|^2 < T$ **then**
5:       $v_n \leftarrow v_n - S_{mn}, n = 1, \ldots, N$
6:    **end if**
7: **end for**
8: **Output:** $\{n : v_n > 0, n = 1, \ldots, N\}$

We further examine one of the major causes of misses, which is that the pulses of two or more neighbors cancel at the receiver, so that the measurement Y_m is below the threshold at multiple intervals. This takes place for two neighbors n_1 and n_2 if their channel coefficients are similar in amplitude but opposite in phase, so that $S_{mn_1} U_{n_1} + S_{mn_2} U_{n_2} \approx 0$ for every interval m where both nodes transmit a pulse, which implies the neighbors will be eliminated erroneously with a number of strikes wherever their pulses coincide.

A simple trick can be used to reduce misses with essentially no impact on false alarm. The idea is to let each node randomize the phases of its signature at different slots independently, i.e., use $S_{mn} e^{j\Theta_{mn}}$ in lieu of S_{mn} where Θ_{mn} are i.i.d. uniform on $[0, 2\pi)$. In this case, if $S_{mn_1} e^{j\Theta_{mn_1}} U_{n_1} + S_{mn_2} e^{j\Theta_{mn_2}} U_{n_2} \approx 0$ for some slot m, it is unlikely that this is still true for other slots. We note that the randomization is easy to implement at transmitters and requires no change at the receivers, because knowledge of the phases is not needed by the noncoherent detection algorithm.
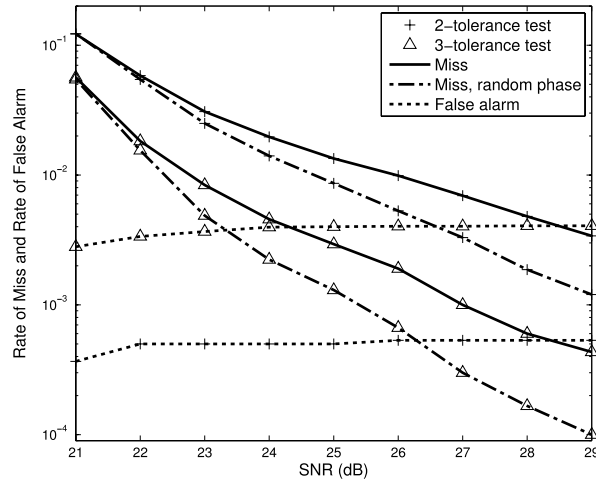
Fig. 2. Rates of miss and false alarm versus SNR. In all 1000 trials, $N = 10,000$, $c = 30$, $M = 2048$, and $q = 0.0176$.

3.3. Design optimization

The simple group testing algorithm and the improved t-tolerance algorithm are in general difficult to analyze. In the absence of noise, there have been asymptotic results (see, e.g., [11,15]) where it is shown that the error probabilities vanish as the problem size increases as long as the number of measurements exceed a certain level, which depends typically logarithmically on the node population. Under a discrete model with noisy measurements, asymptotic performance bounds for the algorithm have been developed in [25]. Other studies of the theoretical limits of noisy measurement models with 0-tolerance test [26–29] assume the active signatures to be transmitted at equal power, and thus do not apply to the current model (1) with fading and path loss.

No existing analytical results or techniques for group testing yields a good approximation of the performance of Algorithm 2. Therefore, we resort to numerical methods to find the optimal design trade-off between cost and error performance. The cost here refers to the signature length (i.e., the neighbor discovery overhead) and the SNR. We assume that the node population and density, the SNR, as well as the fading characteristics are given, which are not controlled by the designer.

For a given signature length M, a good indicator of the rate of miss is Mq, i.e., the average number of active pulses in the signature. Intuitively, for an actual neighbor, the larger Mq is, the more likely its symbols get cancelled by other nodes, thus causing unwanted strikes and misses. On the other hand, if q is too small, some non-neighbors may not receive enough strikes to be eliminated, thereby causing false alarms. Similar qualitative statements can be made about the threshold T used in Algorithms 1 and 2. It is not difficult to see that T should be above the noise variance, but perhaps not too much higher than that.

In this work, we assume the signature length M is given and fixed. We then numerically search the optimal choice of the sparsity q and threshold T as those that minimize the total rate of miss and false alarm. Since using identical signatures under all channel conditions is preferable in practice, the numerical search is carried out under a specific SNR. The same parameters q and T are then used at all other SNRs.

3.4. Numerical results

We show some numerical results obtained based on the design described in Section 3.3.

Suppose there are $N = 10,000$ valid NIAs which belong to nodes uniformly distributed in a square centered at the origin. Let the path loss exponent be $\alpha = 3$. Assume Rayleigh fading and that a node is regarded as a neighbor if the channel gain exceeds $\eta = 0.05$. In each network realization, we consider the average neighbor discovery performance of the 100 nearest nodes to the origin.

Let the density of the network be that there are on average $c = 30$ nodes in each neighborhood. Each signature consists of $M = 2048$ symbols. At 28 dB SNR, the optimal sparsity and threshold are found to be $q = 0.0176$ and $T = 2.0$, respectively, for a 2-tolerance test. The random signature matrix is generated with this fixed sparsity, so that there are on average $Mq = 36$ pulses in a signature. The same signature matrices and threshold are then used at all SNRs in all tests.

The receiver carries out Algorithm 2 and the resulting rates of miss and false alarm are plotted against the SNR in Fig. 2. The rate of false alarm is plotted in dotted lines, the rates of miss with and without random-phase improvement are plotted in dash-dotted lines and solid lines, respectively. The performance of the 2-tolerance test is marked with '+' and that of the 3-tolerance test is marked with '$\triangle$'.

In the case of the 2-tolerance test, missed neighbors are the dominant source of error. Using random phases improves the rate of miss significantly. The rate of miss decreases with the SNR and drops to 0.1% at 29 dB (with random phases).
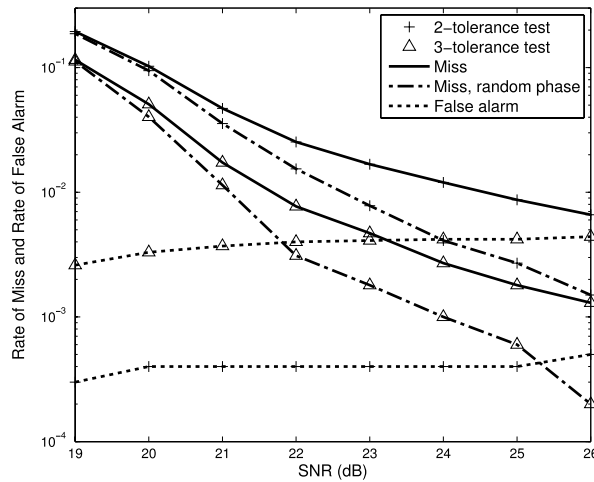
Fig. 3. Rates of miss and false alarm versus SNR. In all 1000 trials, $N = 10{,}000$, $c = 10$, $M = 1024$, and $q = 0.0371$.

The rate of false alarm is not sensitive to the SNR and stays around 0.05%. Using the 3-tolerance test improves the rate of miss significantly (about 4 dB at the error rate of 0.1%), because actual neighbors are less likely to be eliminated. The rate of false alarm, however, becomes higher with higher tolerance. If the total error rate is of concern, then the 2-tolerance test is preferable if the SNR is above 27 dB, whereas the 3-tolerance test is preferable otherwise.

Fig. 3 repeats the preceding experiment, except for a sparser network, where the average number of neighbors a node has is $c = 10$, out of $N = 10{,}000$ nodes, and that shorter signatures are used: $M = 1024$. The parameters $q = 0.0371$ and $T = 3.0$ are optimized for 26 dB for the 2-tolerance test and then used at all SNRs and tests. On average 38 pulses are found in a signature.

Fig. 3 shows that using the 3-tolerance test yields better error rates at high SNRs. At 26 dB, the total error rate is just below 0.1%.

It appears that the improvement from using phase randomization is more pronounced with higher tolerance. In the case of 2-tolerance tests, the rate of miss can be reduced by about 10 fold at high SNRs.

3.5. Network-wide neighbor discovery

Although the preceding development assumes a single query node in the network, it is easy to extend the algorithms to network-wide neighbor discovery, where all or any subset of nodes acquire their neighborhoods simultaneously. This is an advantage of using on–off signatures, because a node can receive useful signals during its own off-slots despite of the half-duplex constraint. In fact the signatures are often very sparse (e.g., $q < 0.04$ in the preceding numerical examples), so that "erased" received symbols due to one's own transmission are few. This also implies that even if the energy of a pulse leaks into neighboring symbol intervals, there are still enough off-slots for making observations.

The impact of the half-duplex constraint is in effect a reduction of the length of the signatures. From the viewpoint of any query node, once the erasures are purged, models (1) and (2) still apply, if the number of measurements M is replaced by a random variable of binomial distribution with parameters $(M, 1 - q)$. For large M, the number of useful measurements is approximately $M(1 - q)$. The discovery algorithm can be carried out by all nodes simultaneously. If we increase M by a factor of $1/(1 - q)$, then the performance of network-wide neighbor discovery is roughly the same as in the case of a single query node with the original signature length.

3.6. Computational complexity

After turning the measurements $\boldsymbol{Y}$ into a binary N-vector by comparing it with a threshold, all computations carried out by Algorithms 1 and 2 are binary or counting down by 1. The computational complexity is $\mathcal{O}(NMq)$ if implemented in a clever way using the sparsity of the signature matrix. If network-wide neighbor discovery is carried out, the complexity at each decoder is increased by a factor of $1/(1 - q) \approx 1 + q$, since q is typically a very small number. A general purpose processor may handle up to $N = 10^5$ NIAs in real time (where M is typically a few thousand). Hardware implementation using, for example, a programmable gate array, may take advantage of the fact that the elimination procedure can be carried out in parallel. In this case, it is conceivable to carry out compressed neighbor discovery for a large address space including all 32-bit Internet Protocol (IP) addresses.

An alternative, more scalable approach proposed in [15] is to divide the address into smaller segments (e.g., a 32-bit address consists of three overlapping 16-bit subaddresses), and discover the subaddresses of all neighbors separately using the preceding algorithms. The subaddresses can then be pieced together to form full addresses by matching their overlaps.

A natural question to ask is why noncoherent group testing algorithms are proposed in this paper in lieu of coherent detection, such as matched filtering followed by thresholding, which should perform better. The reason is that even simple matched filtering entails a much higher complexity with $\mathcal{O}(NM)$ additions over the precision of the measurements.

The problem of inferring about the inputs to a noisy linear system from the outputs has been studied in many contexts. One important area relevant to the model (2) is multiuser detection. Refs. [30,31] consider a related user activity detection problem in cellular networks, and suggest the use of coherent multiuser detection techniques. Such techniques do not apply here because they require knowledge of the channel coefficients U_n of all neighbors, which is clearly unavailable before the neighbors are even known. Ref. [32] considers channel estimation, but the algorithm is more complex than matched filtering, and thus does not scale well with the network size.

The idea of using a t-tolerance test in Algorithm 2 is related to the wisdom of *belief propagation*, where the decision for each node at question is made using beliefs provided by all relevant measurements. One can in fact carry out belief propagation fully and iteratively [33], but we suspect the performance gain does not justify the additional complexity here.

As long as random signatures are used, any good decoding algorithm needs to visit every signature, so that the complexity is at least linear in the address space size N. This prohibits scaling to a very large space, say $N = 2^{48}$. Although random signatures perform as good as any signatures according to Shannon's random coding argument, it is well-known that structures need to be introduced in the codebook in order for low-complexity decoding. This is the subject of the next section.

4. On–off Reed–Muller signatures and chirp decoding

In this section, we propose to use deterministic signatures obtained from second-order Reed–Muller codes with erasures, where the complexity of the corresponding chirp decoding algorithm is sub-linear in N. We first discuss the original RM code without erasure. Such a code is sufficient for a single silent query node to acquire its neighborhood. The construction of the RM code is described in detail in [34]. We provide a sketch of the construction in Section 4.1. The signatures consist of QPSK entries, which prevent a transmitting node from simultaneously discovering its neighborhood. In Section 4.2, zero entries are introduced by erasing about 50% of the symbols in each signature, so that full-duplex neighbor discovery is enabled. The chirp decoding algorithm is discussed in Section 4.3. As we shall see in Section 4.4, using the Reed–Muller code enables more reliable and efficient discovery in networks which are many orders of magnitude larger than allowed by using random on–off signatures.

For the reader's convenience, the signature generation and chirp decoding procedures are summarized as Algorithms 3 and 4. Examples in the case of very small systems are given to illustrate the encoding and decoding procedures.

4.1. The Reed–Muller code (without erasure)

RM codes are a family of linear error-control codes. A formal description of RM codes requires a substantial amount of preparation in finite fields. In a general form, RM codes are based on evaluating certain primitive polynomials in finite fields. Due to space limitations, we briefly describe the second-order RM codes used in this paper using the minimum amount of formalisms. The reader is referred to [34] for a more detailed discussion.

Given a positive integer m, we show how to generate up to $2^{m(m+3)/2}$ distinct codewords, each of length 2^m. For example, in the case of $m = 10$, there are up to 2^{65} codewords of length 1024.

Let $e_l^i = [0, \ldots, 0, 1, 0, \ldots, 0]$ be a row vector of length l in which the i-th entry is equal to 1 whereas all other entries are zeros. Let $\boldsymbol{P}(e_l^i)$ be the $l \times l$ symmetric matrix in which the top row is e_l^i and each of the remaining reverse diagonals (a diagonal from upper right to lower left) is computed from a fixed linear combination of the entries in the top row. The reader is referred to [34] for a detailed description of the construction, which is based on evaluating some primitive polynomials in GF(2^m). For example, $\boldsymbol{P}(e_1^1) = 1$ and

$$\boldsymbol{P}(e_2^1) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad \boldsymbol{P}(e_2^2) = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}. \tag{7}$$

Given m, we form a linear space of $m \times m$ symmetric matrices with a set $\boldsymbol{B}$ of $m(m + 1)/2$ bases constructed using $\{\boldsymbol{P}(e_l^i), i \leq l, l = 1, \ldots, m\}$, where for $l < m$, $P(e_l^i)$ is padded to an $m \times m$ matrix, where the lower right $l \times l$ submatrix is $\boldsymbol{P}(e_l^i)$ and all remaining entries are zeros. In the simple case of $m = 2$, $\boldsymbol{B}$ consists of $m(m + 1)/2 = 3$ bases, which are $\boldsymbol{P}(e_2^1)$, $\boldsymbol{P}(e_2^2)$ given by (7) and an additional matrix obtained from $\boldsymbol{P}(e_1^1) = 1$ by padding zeros:

$$\boldsymbol{B} = \left\{ \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \right\}. \tag{8}$$

Let $\boldsymbol{B}(i)$ denote the i-th basis in $\boldsymbol{B}$ ordered as $\boldsymbol{P}(e_m^1), \ldots, \boldsymbol{P}(e_m^m)$ and then those obtained from $\boldsymbol{P}(e_{m-1}^1), \ldots, \boldsymbol{P}(e_{m-1}^{m-1})$ and so on.

Table 1
16 Reed–Muller codewords.

b	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
c	0	0	0	0	1	1	1	1	2	2	2	2	3	3	3	3
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
$\phi_{\mathbf{b},\mathbf{c}}$	1	-1	1	-1	j	$-j$	j	$-j$	j	$-j$	j	$-j$	1	-1	1	-1
	1	1	-1	-1	1	1	-1	-1	j	j	$-j$	$-j$	j	j	$-j$	$-j$
	1	-1	-1	1	$-j$	j	j	$-j$	-1	1	1	-1	$-j$	j	j	$-j$

Let the NIA consist of $n = n_1 + n_2$ bits, where $n_1 \leq m$ and $n_2 \leq m(m + 1)/2$. Each n-bit NIA is divided into two binary vectors: $\mathbf{b}' \in \mathbb{Z}_2^{n_1}$ and $\mathbf{c} \in \mathbb{Z}_2^{n_2}$, where $\mathbb{Z}_2 = \{0, 1\}$. Let $\mathbf{b} \in \mathbb{Z}_2^m$ be formed by appending $m - n_1$ zeros after $\mathbf{b}'$ ($\mathbf{b} = \mathbf{b}'$ if $n_1 = m$). We map $\mathbf{c}$ to an $m \times m$ symmetric matrix according to

$$\mathbf{P}(\mathbf{c}) = \sum_{i=1}^{n_2} c_i \mathbf{B}(i) \bmod 2 \tag{9}$$

where c_i denotes the i-th bit of $\mathbf{c}$. The corresponding codeword is of 2^m symbols, whose entry indexed by $\mathbf{a} \in \mathbb{Z}_2^m$ is given by

$$\phi_{\mathbf{b},\mathbf{c}}(\mathbf{a}) = \exp\left[j\pi \left(\frac{1}{2}\mathbf{a}^{\mathsf{T}}\mathbf{P}(\mathbf{c})\mathbf{a} + \mathbf{b}^{\mathsf{T}}\mathbf{a} \right) \right]. \tag{10}$$

For example, in the case $m = 2$, there are up to $2^{m(m+3)/2} = 32$ codewords of length $2^m = 4$. Moreover, if the number of nodes is 16, i.e., $n = 4$, only 16 codewords are generated as functions of ($\mathbf{b}, \mathbf{c}$) and given as column vectors in Table 1, where only the first two bases in (8) are used as $n_1 = n_2 = 2$.

4.2. Generation of on–off signatures

The drawback of using the original RM code is that the codewords defined by (10) consist of QPSK symbols, so that a node cannot simultaneously receive useful signals while transmitting its own codeword. In order to achieve full-duplex neighbor discovery, we propose to erase about 50% of the entries of each codeword to obtain an on–off signature, so that nodes can listen during their own off-slots. The signature of each node consists of roughly as many off-slots as on-slots, thus two nodes can receive pulses from each other over about 25% of the slots.

For reasons to be explained shortly in conjunction with the chirp decoding algorithm, we apply random erasures to the signatures in the following simple manner: Suppose n_2 is chosen such that the $m \times m$ symmetric matrix generated by each node is determined by its first $m_0 \leq m/2$ rows. For node k, the erasure pattern $\mathbf{r}_k$ of length 2^m is constructed as follows: Divide $\mathbf{r}_k$ into 2^{m_0} segments with equal length 2^{m-m_0}, let the first segment consist of i.i.d. Bernoulli random variables with parameter $1/2$ and all remaining segments be identical copies of the first segment. It is easy to see that after introducing erasures in the signatures, the network can still accommodate $2^{m(3m+10)/8}$ nodes. For example, if $m = 10$, we have up to 2^{50} signatures of length 1024.

The procedure for generating the on–off signatures based on the RM code is summarized as Algorithm 3.

Algorithm 3 Signature Generation Algorithm

1: *Input: n*-bit NIA
2: Choose m such that $n = n_1 + n_2$ with $n_1 \leq m$ and $n_2 \leq \frac{m_0}{2}(2m - m_0 + 1)$ where $m_0 \leq m/2$.
3: Divide n-bit NIA into two vectors $\mathbf{b}' \in \mathbb{Z}_2^{n_1}$ and $\mathbf{c} \in \mathbb{Z}_2^{n_2}$. Form $\mathbf{b} \in \mathbb{Z}_2^m$ by appending $m - n_1$ zeros after $\mathbf{b}'$.
4: Generate the original RM code $\phi_{\mathbf{b},\mathbf{c}}$ of length 2^m according to (10).
5: Generate the erasure pattern $\mathbf{r}$ of length 2^m as follows: Let the first segment of 2^{m-m_0} bits be i.i.d. Bernoulli random variables with parameter $1/2$ and repeat the segment 2^{m_0} times to form the 2^m bits of $\mathbf{r}$.
6: *Output:* The on-off signature of length 2^m is the element-wise product of $\phi_{\mathbf{b},\mathbf{c}}$ and $\mathbf{r}$.

4.3. The chirp decoding algorithm

We recall that each node makes observations via the multiaccess channel (1), which is a superposition of its neighbors' signatures subject to fading and noise. An iterative chirp decoding algorithm has been developed in [13] to identify the codewords of the RM code based on their noisy superposition. The general idea is to take the Hadamard transform of the auto-correlation of the signal in each iteration to expose the coefficient of the digital chirps and then cancel the discovered signatures from the signal.

In the case of full-duplex discovery, the original chirp decoding algorithm with some modifications can be applied here for any node (say, node 0) to recover its neighborhood based on the observations through its own off-slots (denoted as $\tilde{\mathbf{Y}}$). The details are provided in Algorithm 4.

Algorithm 4 The chirp decoding algorithm

1: *Input:* received signal $\boldsymbol{Y}$ in (2), signatures of all other nodes $\underline{\boldsymbol{S}}$ and its own erasure pattern $\boldsymbol{r}$.
2: Choose the maximum iteration number $T_{\max}$, the threshold η_0 and the maximum number n_0 of weak nodes discovered till termination.
3: Initialize the residual signal $\boldsymbol{Y}_r$ to the pointwise product of $\boldsymbol{Y}$ and $1 - \boldsymbol{r}$.
4: Initialize the iteration number t to 0, the neighbor set $\boldsymbol{N} = \emptyset$ and the coefficient vector $\boldsymbol{C} = \emptyset$.
5: *Main iterations:*
6: **while** $t \leq T_{\max}$ **do**
7:    **for** $i = 1, 2, \ldots, m_0$ **do**
8:       Compute the pointwise multiplication of the conjugate of $\boldsymbol{Y}_r$ and the shift of $\boldsymbol{Y}_r$ in the amount of 2^{m-i}.
9:       Compute the fast Walsh–Hadamard transform of the computed auto-correlation.
10:       Find the position of the highest peak in the frequency domain and decode the i-th row of an $m \times m$ matrix $\boldsymbol{P}(\boldsymbol{c}_k)$, which corresponds to a certain node k.
11:    **end for**
12:    Use the first m_0 rows of the preceding $\boldsymbol{P}(\boldsymbol{c}_k)$ to determine its remaining rows.
13:    Compute $\boldsymbol{S}_k^0(\boldsymbol{a}) = \exp\left[j\pi\left(\frac{1}{2}\boldsymbol{a}^\top \boldsymbol{P}(\boldsymbol{c}_k)\boldsymbol{a}\right)\right]$ for all $\boldsymbol{a} \in \mathbb{Z}_2^m$ and apply Hadamard transform to the pointwise product of $\boldsymbol{Y}_r$ and the conjugate of $\boldsymbol{S}_k^0$;
14:    Recover $\boldsymbol{b}_k$ by finding the highest peak in the frequency domain.
15:    Compute $\phi_{\boldsymbol{b}_k,\boldsymbol{c}_k}$ according to (10) and recover $\boldsymbol{S}_k$ by pointwise product of $\phi_{\boldsymbol{b}_k,\boldsymbol{c}_k}$ and $\boldsymbol{r}_k$.
16:    Add node k to the neighbor set $\boldsymbol{N}$ and add a corresponding 0 to the coefficient vector $\boldsymbol{C}$.
17:    Put together all signatures of nodes in $\boldsymbol{N}$ to form a matrix $\underline{\boldsymbol{S}}_N$. Construct $\tilde{\underline{\boldsymbol{S}}}$ by pointwise multiplying each column in $\underline{\boldsymbol{S}}_N$ with $1 - \boldsymbol{r}$.
18:    Determine the value of vector $\boldsymbol{X}$ which minimizes $\|\boldsymbol{Y}_r - \tilde{\underline{\boldsymbol{S}}}\boldsymbol{X}\|_2$. Update the coefficient vector $\boldsymbol{C}$ by $\boldsymbol{C} + \boldsymbol{X}$.
19:    Update the residual signal $\boldsymbol{Y}_r$ by $\boldsymbol{Y}_r - \tilde{\underline{\boldsymbol{S}}}\boldsymbol{X}$.
20:    **if** $\boldsymbol{N}$ contains more than n_0 nodes with coefficients less than η_0 **then**
21:       Stop the main iteration.
22:    **end if**
23: **end while**
24: *Output:* All elements in $\boldsymbol{N}$ whose corresponding coefficients in $\boldsymbol{C}$ are no less than η_0.

In the following, we provide a simple example to illustrate the key steps of Algorithm 4. Consider a network of $N = 2^n = 1024$ nodes. Let the parameters in Algorithm 3 be $n_1 = 5, n_2 = 5, m = 5, m_0 = 1$, so that we have 1024 signatures of length $2^m = 32$. Suppose for simplicity node 0 has only two neighbors, whose on–off signatures are $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$, respectively:

$$\boldsymbol{S}_1 = [1, 0, 0, 1, 1, 1, 0, 1, 0, j, 0, j, 0, j, -j, j, -j, 0, 0, -j, j, j, 0, j, 0, 1, 0, 1, 0, -1, 1, -1] \tag{11}$$

$$\boldsymbol{S}_2 = [0, 0, j, 0, 0, -j, 1, 0, 0, j, -1, 0, 0, 0, 0, 0, 0, 0, -j, 0, 0, -j, 1, 0, 0, -j, 1, 0, 0, 0, 0, 0] \tag{12}$$

where the zeros in the signatures are due to erasures. Suppose the channel gains are $U_1 = 3$ and $U_2 = 2j$. In the absence of noise, node 0 observes the signal $U_1\boldsymbol{S}_1 + U_2\boldsymbol{S}_2$ through its own off-slots as

$$\tilde{\boldsymbol{Y}} = [3, 0, 0, 3, 0, 0, 0, 0, 0, 0, -2j, 3j, 0, 3j, -3j, 3j, -3j, 0, 0, -3j, 0, 0, 0, 0, 0, 0, 2j, 3, 0, -3, 3, -3]. \tag{13}$$

Given $\boldsymbol{Y}_r$ is initialized to $\tilde{\boldsymbol{Y}}$, the key steps of Algorithm 4 leading to the discovery of the first neighbor is described as follows:

1. Steps 7 to 12:
   Note that $m_0 = 1$ in this case. Take the Hadamard transform of the auto-correlation function of $\boldsymbol{Y}_r$ and its shift by 2^{m-1} to expose the chirps in the frequency domain, so that the first row of $\boldsymbol{P}(\boldsymbol{c})$ can be recovered, and then the entire matrix can be determined. Using $\tilde{\boldsymbol{Y}}$ given by (13), the index of the highest peak is the 21st. Therefore, the first row of $\boldsymbol{P}(\boldsymbol{c})$ is the binary representation of 20, i.e., the binary string of 10100. The matrix $\boldsymbol{P}(\boldsymbol{c})$ can then be uniquely determined.

2. Steps 13 and 14:
   Compute $\boldsymbol{S}^0(\boldsymbol{a}) = \exp\left[j\pi\left(\frac{1}{2}\boldsymbol{a}^\top \boldsymbol{P}(\boldsymbol{c})\boldsymbol{a}\right)\right]$ for all $\boldsymbol{a} \in \mathbb{Z}_2^m$ and apply Hadamard transform to the pointwise product of $\boldsymbol{Y}_r$ and the conjugate of $\boldsymbol{S}^0$ to recover $\boldsymbol{b}$. In the example, the index of the highest peak is the 19-th in the first iteration; hence $\boldsymbol{b} = 10{,}010$.

3. Steps 15 to 18:
   Recover the erased signature $\boldsymbol{S}$ by pointwise product of $\phi_{\boldsymbol{b},\boldsymbol{c}}$ and $\boldsymbol{r}$, then put together all signatures already recovered to form a matrix $\tilde{\underline{\boldsymbol{S}}}$, where all rows of $\tilde{\underline{\boldsymbol{S}}}$ corresponding to the on-slots of node 0 are set to zero. Determine the value of $\boldsymbol{X}$ which minimizes $\|\boldsymbol{Y}_r - \tilde{\underline{\boldsymbol{S}}}\boldsymbol{X}\|_2$. In the example, the reconstructed signature in the first iteration corresponds to the signature of the first neighbor ($\boldsymbol{S}_1$) and the corresponding coefficient X_1 is estimated to be 3, which is equal to U_1.

The preceding steps are repeated to discover more nodes. The algorithm terminates if either the total number of iterations reaches the maximum number of iterations as one desires, or among the discovered nodes, enough of them correspond to very weak coefficients, which implies that the algorithm starts to produce non-neighbors.
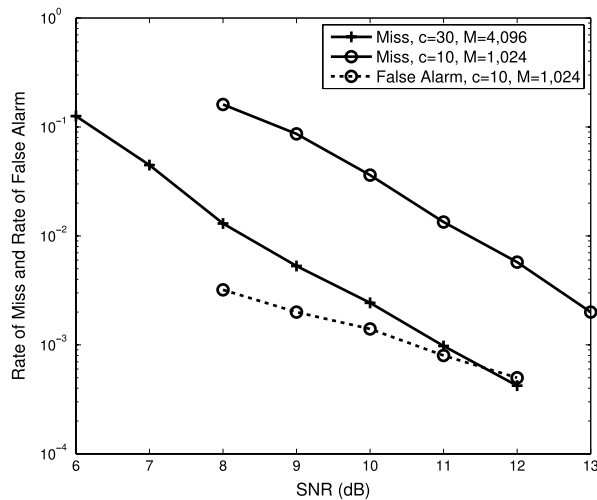
Fig. 4. The rates of miss and the rate of false alarm versus SNR.

We now justify the special scheme for generating the erasures in Algorithm 3. In order to recover the i-th row of the $m \times m$ symmetric matrix corresponding to the largest energy component in the residual signal, the auto-correlation is computed between the residual signal of length 2^m and its shift by 2^{m-i}. It is advisable to guarantee that the positions of erasures in the received signal and its shift are perfectly aligned as designed in Algorithm 3.

4.4. A numerical example

We illustrate the performance of discovery using RM codes through the following example. The same network model is assumed as in Section 3.4, except with a larger set of 2^{20} valid NIAs.

First let the density of the nodes be such that each node has on average $c = 10$ neighbors. Choose $m = n_1 = n_2 = 10$, then the signature length is $2^m = 1024$. Averaged over 10 network realizations of the large network, the rate of miss and the rate of false alarm of a total $10 \times 100 = 1000$ nodes (with approximately 10,000 neighbors in total) are plotted in Fig. 4 against the SNR. Note that there are no false alarms registered during the simulation when SNR is larger than 12 dB. We find that the total error rate can be lower than 0.2% at 13 dB SNR. In contrast, if random on–off 1024-bit signatures are used instead (see Fig. 3), at least 26 dB SNR is needed to achieve the same error rate, even if the size of the address space is only 10,000.

We repeat the simulation with the number of average neighbors changed to $c = 30$ and the parameters changed to $m = n_2 = 12$, and $n_1 = 8$. In this case, the signature length is $2^m = 4096$. During all 10 network realizations, there are no false alarms and the total error rate can be lower than 0.2% at 11 dB SNR.

In order to show that the chirp decoding algorithm is highly resilient to the near–far problem, we demonstrate in Fig. 5 that strong neighbors will be detected with very high probability so that their interference to weaker neighbors can be removed. In the case of average $c = 10$ neighbors, when the signature length is 1024 and SNR is 10 dB, we can see that the rate of miss decreases as the neighbors become stronger, and the rate of miss is below 0.1% at -6 dB attenuation. The simulation is repeated with the number of average neighbors changed to $c = 30$, the length of signature changed to 4096 and SNR changed to 7 dB. We can see that neighbors with attenuation less than -6 dB are successfully discovered with miss rate less than 0.2%.

5. Comparison with random access

We compare the performance of the compressed neighbor discovery schemes described in Sections 3 and 4 with that of conventional random-access discovery schemes. Only one frame interval is needed by compressed neighbor discovery, as opposed to many frames (often in the hundreds) in the case of random access. Thus compressed neighbor discovery also offers significant reduction of synchronization and error-control overhead embedded in every frame.

5.1. Comparison with generic random-access discovery

Suppose a random-access discovery scheme is used, such as the "birthday" algorithm in [2]. Nodes contend to announce their NIAs over a sequence of k contention periods. In each period, each neighbor independently chooses to either transmit
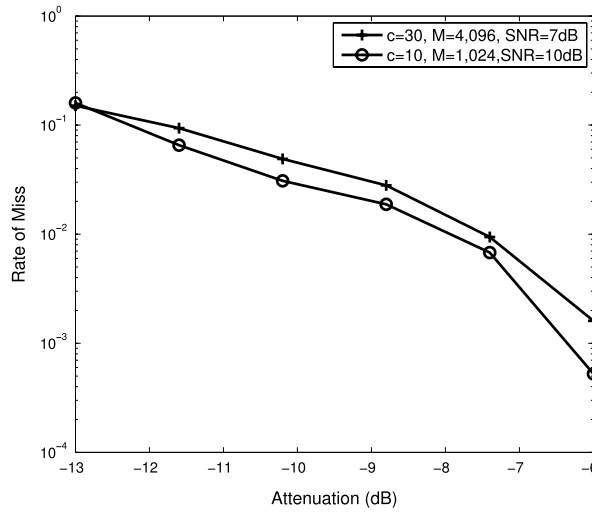
Fig. 5. The rate of miss versus attenuation.

Table 2
Comparison between random-access discovery and compressed neighbor discovery based on group testing (10,000 NIAs).

	Random access	Group testing
$c = 10$	194 frames 1358 symbols	1 frame 1024 symbols
$c = 30$	534 frames 3738 symbols	1 frame 2048 symbols

Table 3
Comparison between random-access discovery and compressed discovery based on RM codes (10^{20} NIAs).

	Random access	RM codes
$c = 10$	194 frames 1940 symbols	1 frame 1024 symbols
$c = 30$	534 frames 5340 symbols	1 frame 4096 symbols

(with probability θ) or listen (with probability $1 - \theta$). Let $\rho = c/N$. The error rate is equal to the probability of one given neighbor being missed, which is given by

$$\sum_{z=1}^{N} \binom{N}{z} \rho^z (1 - \rho)^{N-z} \left[1 - \theta (1 - \theta)^{z-1} \right]^k. \tag{14}$$

Consider a network with 10,000 NIAs, so in each contention period, the number of bits transmitted is at least $\lceil \log_2(10^4) \rceil = 14$ just to carry the NIA. For a fair comparison with compressed neighbor discovery schemes, we assume time is slotted and QPSK modulation is used. Table 2 lists the amount of transmissions needed by random access discovery according to (14) and by compressed discovery based on 2-tolerance group testing (see Figs. 2 and 3) in order to achieve the target error rate of 0.002 in the case of 10 or 30 neighbors.

Evidently, random-access discovery requires hundreds of 14-bit frame transmissions to guarantee the same performance achieved by compressed discovery using a single frame transmission. The latter scheme uses much longer frames. Still, the total number of symbols required by compressed discovery is substantially smaller, and in fact the advantage is greater in the case of more neighbors.

A similar comparison can be made between random-access discovery and compressed discovery based on RM codes. Consider a network with 2^{20} nodes. To achieve the target rate of 0.002 in the case of 10 or 30 neighbors, Table 3 lists the amount of transmissions needed by random-access discovery according to (14) and by compressed discovery based on RM codes with chirp decoding (see Fig. 4). Again, compressed discovery significantly outperforms random-access discovery.

The efficiency of compressed neighbor discovery can be significantly higher than that of random access if all overhead is accounted for. This is because that sending a 14-bit or 20-bit NIA reliably over a fading channel may require up to a hundred symbol transmissions or more. We believe using compressed discovery can reduce the amount of total discovery overhead by an order of magnitude.

5.2. Comparison with IEEE 802.11g

It is also instructive to compare compressed neighbor discovery with the popular IEEE 802.11g technology. Consider the ad hoc mode of 802.11g with active scan, which is basically a random-access discovery scheme. The signaling rate is 4 μs per orthogonal frequency division multiplexing (OFDM) symbol. One probe response frame takes about 850 μs. (The response frame includes additional bits but is dominated by the NIA.) Thus it takes at least 850 μs × 194 ≈ 165 ms for a query node to discovery 10 neighbors with error rate 0.002 or lower. If compressed neighbor discovery with on–off signature is used, 1024 symbol transmissions suffice to achieve the same error rate. Using 802.11g symbol interval (4 μs), reliable discovery takes merely 4.1 ms. A highly conservative choice of the symbol interval is 30 μs, which includes carrier (on–off) ramp period (say 10 μs) and the propagation time (less than 1 μs for 802.11 range). Compressed neighbor discovery then takes a total of 30 ms, less than 1/5 of that required by 802.11g.

6. Concluding remarks

In this paper, we have developed two compressed neighbor discovery schemes, which are efficient, scalable, and easy to implement. The on–off signaling used in neighbor discovery schemes was first proposed in [35] and referred to as *rapid on–off-division duplex*, or *RODD*. Such signaling departs from the collision model and fully exploits the superposition nature of the wireless medium [36]. Moreover, using on–off signatures allows half-duplex nodes to achieve network-wide full-duplex discovery. It is interesting to note that transmission of pulses by each node (which identifies the node) is scheduled at the symbol level, rather than at the timescale of the frame level.

The neighbor discovery problem is different from most other applications of compressed sensing in the literature because of the sheer scale of the problem. The number of unknowns is typically 2^{20} or more. We choose to use RM codes because of its scalability and effectiveness for compressed sensing. At the time of this work, there are no other practical codes which are known to deliver comparable performance for noisy compressed sensing at this scale and efficiency.

A brief discussion of how neighbor discovery is triggered is in order. If a single node (e.g., a new comer) is interested in its neighborhood, it may send a query message, so that only the neighbors which can hear the message will respond immediately. To implement network-wide discovery, nodes can be programmed to simultaneously transmit their on–off signatures at regular, pre-determined epochs, so that all nodes discover their respective neighbors. This also prevents neighbor discovery from interfering with data transmission.

Compressed neighbor discovery is well suited and in fact significantly outperforms existing schemes for mobile networks where the topology of the network changes over time. Depending on the mobility, it may be desirable to carry out neighbor discovery periodically. If this is done frequently, the topology may not change much, hence it is also possible to create a Markov model for connectivity and incorporate the model into the neighborhood inference problem. This is left to future work.

Finally, we note that Qualcomm has developed the FlashLinQ technology based on OFDM, which carries out neighbor discovery over a large number of orthogonal time-frequency slots [37]. Over each slot, however, the scheme is still based on random access. The schemes proposed in this paper can also be extended to multicarrier systems. This is also a direction for future work.

References

[1] RFC 3684: Topology dissemination based on reverse-path forwarding (TBRPF), MANET Working Group, The Internet Engineering Task Force (IETF), 2004.
[2] M.J. McGlynn, S.A. Borbash, Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks, in: Proceedings of the 2nd ACM International Symposium on Mobile Ad hoc Networking & Computing, 2001, pp. 137–145.
[3] S. Vasudevan, J. Kurose, D. Towsley, On neighbor discovery in wireless networks with directional antennas, in: Proc. IEEE INFOCOM 4, 2005, pp. 2502–2512.
[4] E. Felemban, R. Murawski, E. Ekici, S. Park, K. Lee, J. Park, Z. Hameed, SAND: Sectored-antenna neighbor discovery protocol for wireless networks, in: Proc. IEEE Conf. Sensor Mesh and Ad Hoc Communications and Networks, 2010, pp. 1–9.
[5] S. Vasudevan, D. Towsley, D. Goeckel, R. Khalili, Neighbor discovery in wireless networks and the coupon collector's problem, in: Proc. ACM Mobicom, Beijing, China, 2009, pp. 181–192.
[6] R. Khalili, D. Goeckel, D. Towsley, A. Swami, Neighbor discovery with reception status feedback to transmitters, in: Proc. IEEE INFOCOM, San Diego, CA, USA, 2010, pp. 1–9.
[7] S.A. Borbash, A. Ephremides, M.J. McGlynn, An asynchronous neighbor discovery algorithm for wireless sensor networks, Ad Hoc Networks 5 (2007) 998–1016.
[8] D.L. Donoho, Compressed sensing, IEEE Trans. Inform. Theory 52 (2006) 1289–1306.
[9] E.J. Candes, T. Tao, Near-optimal signal recovery from random projections: universal encoding strategies? IEEE Trans. Inform. Theory 52 (2006) 5406–5425.
[10] D.-Z. Du, H.K. Hwang, Combinatorial Group Testing and its Applications, second ed., in: Series on Applied Mathematics, vol. 12, World Scientific, Singapore, 1993.

[11] T. Berger, N. Mehravari, D. Towsley, J. Wolf, Random multiple-access communication and group testing, IEEE Trans. Commun. 32 (1984) 769–779.
[12] M. Sudan, Coding theory: Tutorial & survey, in: Proc. of the 42nd Annual Symposium on Foundations of Computer Science, 2008, pp. 11–15.
[13] S.D. Howard, A.R. Calderbank, S.J. Searle, A fast reconstruction algorithm for deterministic compressive sensing using second order Reed–Muller codes, in: Proc. Conf. Inform. Sciences & Systems, 2008, pp. 11–15.
[14] L. Zhang, D. Guo, Neighbor discovery in wireless networks using compressed sensing with Reed–Muller codes, in: Proc. WiOpt, Princeton, NJ, USA, 2011.
[15] J. Luo, D. Guo, Neighbor discovery in wireless ad hoc networks based on group testing, in: Proc. Allerton Conf. Commun., Control, and Computing, Monticello, IL, USA, 2008.
[16] J. Luo, D. Guo, Compressed neighbor discovery for wireless ad hoc networks: the Rayleigh fading case, in: Proc. Allerton Conf. Commun., Control, and Computing, Monticello, IL, USA, 2009.
[17] Y. Wu, S. Verdú, Rényi information dimension: Fundamental limits of almost lossless analog compression, IEEE Trans. Inform. Theory 56 (2010) 3721–3748.
[18] O. Simeone, U. Spagnolini, Y. Bar-Ness, S. Strogatz, Distributed synchronization in wireless networks, IEEE Signal Processing Mag. 25 (2008) 81–97.
[19] F. Baccelli, B. Błaszczyszyn, Stochastic Geometry and Wireless Networks: Volume I Theory and Volume II Applications, vol. 4, NoW Publishers, 2009.
[20] T.S. Rappaport, Wireless Communications, second ed., Prentice-Hall, 2002.
[21] S. Ramanathan, A unified framework and algorithm for channel assignment in wireless networks, Wireless Networks 5 (1999) 81–94.
[22] S. Xu, T. Saadawi, Does the IEEE 802.11 MAC protocol work well in multihop wireless ad hoc networks? IEEE Communication Magazine 39 (2001) 130–137.
[23] M. Kodialam, W.C. Lau, T. Nandagopal, Identifying RFID tag categories in linear time, in: Proc. WiOpt, Seoul, Korea, 2009.
[24] D. Kurtz, M. Sidi, Multiple access algorithms via group testing for heterogeneous population of users, IEEE Trans. Commun. 36 (1988) 1316–1323.
[25] E. Knill, W.J. Bruno, D.C. Torney, Non-adaptive group testing in the presence of errors, Discrete Appl. Math. 88 (1998) 261–290.
[26] M. Cheraghchi, A. Hormati, A. Karbasi, M. Vetterli, Compressed sensing with probabilistic measurements: a group testing solution, in: Proc. Allerton Conf. Commun., Control, and Computing, 2009, pp. 30–35.
[27] M. Cheraghchi, A. Karbasi, S. Mohajer, V. Saligrama, Graph-constrained group testing, in: Proc. IEEE Int. Symp. Inform. Theory, 2010, pp. 1913–1917.
[28] M. Cheraghchi, Derandomization and group testing, in: Proc. Allerton Conf. Commun., Control, and Computing, 2010, pp. 991–997.
[29] G. Atia, V. Saligrama, Boolean compressed sensing and noisy group testing, 2010, arXiv:0907.1061v4.
[30] D.D. Lin, T.J. Lim, Subspace-based active user identification for a collision-free slotted ad hoc network, IEEE Trans. Commun. 52 (2004) 612–621.
[31] D. Angelosante, E. Biglieri, M. Lops, A simple algorithm for neighbor discovery in wireless networks, in: Proc. IEEE Int'l Conf. Acoustics, Speech and Signal Processing, vol. 3, pp. 169–172.
[32] D. Angelosante, E. Biglieri, M. Lops, Neighbor discovery in wireless networks: A multiuser-detection approach, Physical Communication 3 (2010) 28–36.
[33] F.R. Kschischang, B.J. Frey, H.-A. Loeliger, Factor graphs and the sum–product algorithm, IEEE Trans. Inform. Theory 47 (2001) 498–519.
[34] A.R. Calderbank, A.C. Gilbert, M.J. Strauss, List decoding of noisy reed-muller-like codes, CoRR, 2006.
[35] D. Guo, L. Zhang, Rapid on–off-division duplex for mobile ad hoc networks, in: Proc. Allerton Conf. Commun., Control, and Computing, Monticello, IL, USA, 2010.
[36] L. Zhang, D. Guo, Capacity of Gaussian channels with duty cycle and power constraints, in: Proc. IEEE Int. Symp. Inform. Theory, St Petersburg, Russia, 2011.
[37] J. Ni, R. Srikant, X. Wu, Coloring spatial point processes with applications to peer discovery in large wireless networks, in: SIGMETRICS, 2010, pp. 167–178.

Lei Zhang received the B.E. and M.S. degrees from the Department of Electronics Engineering, Tsinghua University, Beijing, China, in 2005 and 2007, respectively. He is currently pursuing the Ph.D. degree in the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL. His research interests include wireless ad hoc networks, compressed sensing and information theory. He is a recipient of the student paper award of IEEE International Symposium on Information Theory in 2011. He also received Walter P. Murphy Fellowship and Terminal Year Fellowship from Northwestern University in 2007 and 2011, respectively.

Jun Luo received the M.Sc. and Ph.D. degrees in Electrical Engineering from Northwestern University, Evanston, IL, and the B.Sc. degree in Mathematics from Peking University, Beijing, China. His research was focused on information theory and wireless ad hoc networks. He is currently with Goldman Sachs.
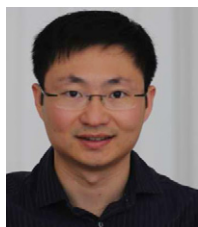
Dongning Guo joined the faculty of Northwestern University, Evanston, IL, in 2004, where he is currently an Associate Professor in the Department of Electrical Engineering and Computer Science. He received the B.Eng. degree from the University of Science & Technology of China, the M.Eng. degree from the National University of Singapore, and the M.A. and Ph.D. degrees from Princeton University, Princeton, NJ. He was an R&D Engineer in the Center for Wireless Communications (now the Institute for Infocom Research), Singapore, from 1998 to 1999. He has held visiting positions at Norwegian University of Science and Technology in summer 2006 and in the Institute of Network Coding at the Chinese University of Hong Kong in 2010–2011. He is an Associate Editor of the IEEE Transactions on Information Theory and an Editor of Foundations and Trends in Communications and Information Theory.

Dongning Guo received the Huber and Suhner Best Student Paper Award in the International Zurich Seminar on Broadband Communications in 2000 and is a co-recipient of the IEEE Marconi Prize Paper Award in Wireless Communications in 2010 (with Y. Zhu and M.L. Honig). He is also a recipient of the National Science Foundation Faculty Early Career Development (CAREER) Award in 2007. His research interests are in information theory, communications, and networking.