Simultaneous Query for Wireless Sensor Networks: A Power Based Solution

Dingming Wu, *Student Member, IEEE*, Guihai Chen, *Member, IEEE*, Chao Dong, *Member, IEEE*, Shaojie Tang, *Member, IEEE*, and Haipeng Dai, *Member, IEEE*

Abstract—We study the problem of supporting simultaneous query in WSNs. For such networks, the network efficiency can be significantly improved if the poller can obtain information via a simultaneous query. Two fundamental cases of such queries are studied in this paper: counting and identifying active neighboring nodes. We propose two mechanisms, *Power based counting* (Poc) and *Power based identification* (Poid), which achieve the goals by allowing neighbors to respond simultaneously to a poller. A key observation that motivates our design is that the power of a superposed signal increases with the number of component signals under the condition that the component signals are synchronized precisely. However, such high precision of synchronization is rather difficult to achieve in WSNs. To address this challenge, we design delicate delay compensation methods to reduce the phase offset of each component signal. Moreover, we propose a novel probabilistic estimation technique to overcome the hardware limitations on the observed received power. Poid currently works only in sparse networks though, we discuss and analyze the time complexity of applying Poid in dense networks. Experimental results show that the average accuracy of Poc and Poid is above 95 and 91 percent, respectively. In addition, our methods achieve substantially lower energy consumption and estimation delay compared with the state-of-the-art solutions.

Index Terms—Simultaneous query, counting, identification, constructive interference, wireless sensor networks

1 Introduction

Wireless sensor networks (WSNs) bring revolutionary changes to various applications, e.g., environment monitoring, industrial process control, and battlefield surveillance. In such applications, a poller node may usually need to be aware of their neighbors' up-to-date states to perform decision making. Counting the number of neighbors or identifying neighbors' identities that satisfy a certain condition are two typical cases. For example, nodes may be interested in the following: (1) how many of my neighbors have a battery level above a certain threshold [1]; (2) did anyone lost my broadcasting message [2]; (3) which nodes who have witnessed a common event [3].

To obtain such information, the poller needs to adopt a polling-based scheme which queries each neighbor sequentially, or a TDMA-based strategy which reserves an unique time slot for each neighbor to respond in a fixed order [4]. While simple and lightweight, these approaches are not efficient since the poller's communication delay and energy consumption increase linearly with the network size. Recently, Zeng et al. [5] proposed two RSSI-based counting schemes, LinearPoll and LogPoll, which allow neighbors to respond simultaneously. However, LinearPoll has only constant energy consumption improvement over the TDMA-based scheme and LogPoll counts nodes only on a logarithmic scale.

Different from the existing works, we propose to utilize constructive interference (CI) to tackle the counting and identification problem. CI is a new trend in wireless communications and has been leveraged to achieve fast network flooding and time synchronization [6] or fast data dissemination [7]. In this work, we take advantage of some nice properties of the received power of superposed signals under CI. A key insight behind our design is that the received power of a superposed signal is predictable with bounded error if the received power of each component signal can be accurately predicted, and the phase offset (PO) of each component signal has bounded variability. We construct a power prediction model by following this insight. Based on this model, we propose two novel mechanisms, *Power based counting* (Poc) and *Power based identification* (Poid), which realize fast and fine-grained counting and identification in static networks. Both of these two mechanisms assign each neighbor a responding power in an offline manner. Nevertheless, Poc and Poid adopt different power assignment schemes. In particular, Poid assigns diverse responding power for each neighbor and ensures that the received powers from any two different neighbor sets are sufficiently different. Poid then estimates the set of responders by identifying the unique received power. Conversely, Poc ensures that the received powers from all neighbors are similar, and counts nodes by identifying the power gain of superposed signals. Poc adopts a novel probabilistic estimation technique and is resilient to the network density. Since received power information is processed locally, both these two mechanisms have constant communication delay, thus achieving fast counting and identification with constant energy cost.

- *D. Wu, G. Chen, and H. Dai are with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China. E-mail: {dmwu0506, hpphd2003}@gmail.com, gchen@nju.edu.cn.*
- *C. Dong is with the Department of Communication Engineering, PLA University of Science and Technology, Nanjing 210007, China. E-mail: dch999@gmail.com.*
- *S. Tang is with the Department of Information Systems, University of Texas at Dallas, Dallas, TX 75080. E-mail: tangshaojie@gmail.com.*

Manuscript received 16 June 2014; revised 10 Mar. 2015; accepted 15 Mar. 2015. Date of publication 24 Mar. 2015; date of current version 4 Jan. 2016. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TMC.2015.2416173

In this paper, we first identify two grand challenges faced in our design. First, while the received power from individual neighbors can be accurately predicted in static networks [5], power superposition from multiple neighbors is rather difficult to predict. This is because the POs across different transmissions are quite random. To handle this challenge, we deeply investigate different types of delays caused by nodes' radios and MCUs. We show that by compensating the signal propagation delay across different nodes in an offline manner, POs can be reduced to a level of 0.25 μs with very small variability. The second challenge is that there exists an upper bound on the observed received power due to the poller's hardware limitations, e.g., analog-to-digital converter (ADC) saturation. Consequently, the observed received power of a superposed signal cannot truly reflect its true value if it's too large. To address this challenge for Poc, we assign a response probability p to each node, and let nodes respond with probability p to ensure that the received power is below the upper bound with high probability. We present methods to find the optimal p and develop a novel two-phase estimation to reduce the estimation delay and improve the estimation accuracy. In Poid, we discuss the time-slots method augmented with the ability to count multiple responses in a single slot, and show that under this condition, the number of time slots can be reduced. We also derive the upper bound and lower bound on the number of slots required by Poid.

The major contributions of this paper can be summarized as follows. (1) We design signal propagation delay compensation approaches to synchronize concurrent transmissions with very high precision. (2) We develop the model of predicting power of individual signals and superposed signals. Experimental results show that our model is reliable when the received power is upper bounded by a certain value. (3) Based on the power model, we propose Poc and Poid: Poc achieves fast and fine-grained counting and is resilient to network topology; Poid achieves accurate node identification with constant delay. (4) We implement these two schemes on a testbed of 1 USRP1 [8] and 50 TelosB [9] nodes. Experimental results show that the accuracy of Poc is above 95 percent, and the accuracy of Poid exceeds 91 percent for most cases. In addition, our methods achieve substantially lower estimation delay compared with the state-of-the-art solutions.

2 Superposed Signal with CI

Constructive interference states the ability for a common receiver to decode multiple concurrent packet transmissions as long as these packets are identical and superimposed each other with very little phase offsets. The original work in Glossy [6] show that the phase offset should be no larger than 0.5 μs in IEEE 802.15.4 radios. A key insight behind our design is that the received power of a superposed signal increases with the number of responders if the component signals are added to each other constructively. To better understand such relationship, we provide theoretical amplitude analysis and conduct preliminary experiments. Experimental results reveal the limitations of power superposition under CI, which trigger our further consideration on the design of Poc and Poid.

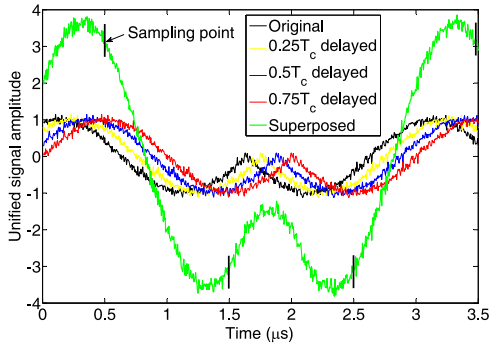


Fig. 1. Waveform of a superposed signal and four component signals.

2.1 Signal Power under CI

The IEEE 802.15.4 standard [10] compatible radios equipped in sensor nodes adopt an offset quadrature phase-shift keying (O-QPSK) modulation scheme. By O-QPSK modulation, the quadrature phase signal is delayed by $T_c = 0.5\,\mu$s with respect to the in-phase signal. To achieve CI in sensor networks, the maximum phase offset (MPO) across different transmissions must be no larger than 0.5 μs [6].

Consider multiple signals superpose at the poller, let $s_r(t)$ denote the received superposed signal from n responders, A_i, τ_i, n_i are the amplitude, phase offset and noise of the ith signal. $s_r(t)$ can be calculated as $s_r(t) = \sum_{i=1}^{n}(A_i s(t-\tau_i) + n_i(t))$. After performing coherent demodulation on $s_r(t)$, the poller will sample the baseband signals with period $2T_c$. According to [11], the amplitude A_r of the received signal can be expressed as $A_r = \sum_{i=1}^{n}(A_i \cos(\frac{\pi}{2T_c}\tau_i) + n_i)$, and the received power P_r is

$$P_r = A_r^2 = \left[\sum_{i=1}^{n}\left(A_i \cos\left(\frac{\pi}{2T_c}\tau_i\right) + n_i\right)\right]^2. \quad (1)$$

In Fig. 1, a four-chips ([1 0 0 1]) signal and three replicas with phase offsets [0.25 0.5 0.75] T_c are plotted. All have the same unit amplitude ($A_i = 1$) and white Gaussian noise with power level being 0.01. The superposed signal is demonstrated with marks on the sampling points where $t = kT_c$ ($k = 1, 3, 5...$). As can be seen, the amplitude of the superposed signal after sampling is about $\sum_{i=1}^{4} \cos(\frac{\pi}{2T_c} \cdot \tau_i) = 3.01$, which is much larger than the original signal.

2.2 Signal Synchronization

In [6], Ferrari et al. showed that by compensating software processing delays and mitigating hardware delays, CI can be achieved in commodity sensor platforms with CC2420 radios. However, CI alone cannot ensure sufficient power gain of the superposed signals. Consider an extreme case where all delayed signals have phase offsets of 0.5 μs, P_r would be equal to the power of a single signal. In the following, we show that by further compensating the signal propagation delay, we can achieve a higher level of synchronization in WSNs.

2.2.1 Compensating Signal Propagation delay

We first take a closer look at the delays of different nodes that lead to the phase offsets among concurrent signals.

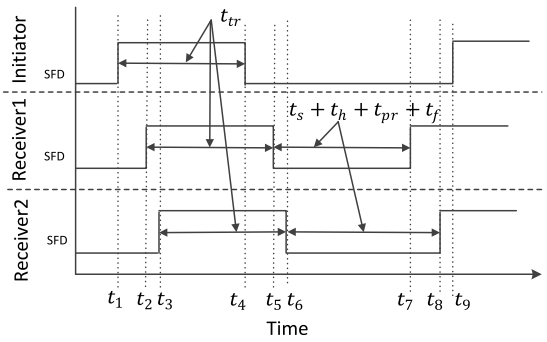


Fig. 2. Execution timeline of an initiator and two receivers with respect to the transitions of SFD pins.

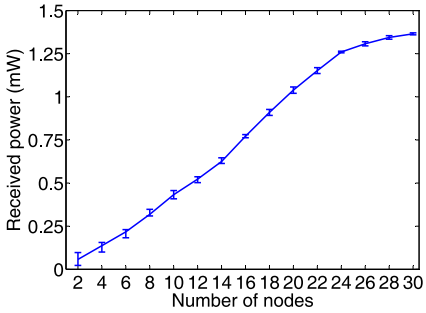


Fig. 3. Received power against different number of simultaneous transmissions.

Signal propagation delay t_p is the duration of signal's two-way around flight time between an initiator and a receiver. Strictly, there also exists a delay in the radio's sensing an arriving transmission (i.e., the time required by radios to successfully decode the first symbol). However, this delay is hard to evaluate. Here, we just consider the sum of propagation delay and radio's sensing delay as the signal propagation delay.

Software delay t_s is the sum of (1) the delay that microcontroller unit (MCU) detects the transition of the radio's SFD pin after the radio signaling its completion of a packet reception, (2) the number of MCU clocks before it issues a transmission request to the radio, (3) the delay that radios detect the transmission request from the MCU.

Hardware delay t_h describes the time required by radios to calibrate their internal voltage controlled oscillators (VCO) to switch from packet reception state to transmission state.

In Fig. 2, we plot the SFD activities of one initiator and two receivers and specify the above three delays using the transitions of SFD pin as common references. Here, t_{pr} and t_f denote the time duration required by radios to transmit the preamble and the start of frame delimiter of a packet, respectively. These two delays only depend on the standard. Since software delay t_s can be accurately evaluated and compensated and hardware delay t_h can be mitigated by issuing a transmission request before reading the whole packet out from the Rx buffer [6], we assume that $t_s + t_h + t_{pr} + t_f$ is fixed for all nodes. Hence the signal propagation delay of receiver 1 and 2 can be expressed as $t_{p1} = (t_9 - t_1) - (t_7 - t_2), t_{p2} = (t_9 - t_1) - (t_8 - t_3)$, where $t_9 - t_1$ is the duration between two rising edges of the SFD pin on the poller and so on to the receivers. We denote this duration of the initiator and receiver i as t_c^0 and t_c^i respectively. Thus the signal propagation delay of receiver i can further be expressed as $t_{pi} = t_c^0 - t_c^i$. According to the experimental results, t_{p1} differs from t_{p2} significantly, even if the two receivers share a similar distance to the initiator. This is due to the clock frequency drifts among different nodes.

Based on the fact that all nodes have the same data transmission delay (denoted as t_{tr} in Fig. 2), we can estimate the clock drift coefficient of each node relative to the initiator [11]. Define by λ_i the clock drift coefficient of node i relative to the initiator. Then in Fig. 2 we have $\lambda_1 = \frac{t_4 - t_1}{t_5 - t_2}$, $\lambda_2 = \frac{t_4 - t_1}{t_6 - t_3}$. To estimate λ_i, the initiator first sends a packet to node i and records the instants of transitions of its SFD pin. On receiving the packet, node i also records the instants of transitions of its SFD pin and piggybacks these instants to the initiator. The initiator thereby computes λ_i as in the previous equations. After obtaining the clock drift coefficient of each node, the initiator calibrates its estimation of the signal propagation delay of node i: $t'_{pi} = t_c^0 - t_c^i \lambda_i$. This process can be repeated for multiple rounds and get the average t'_{pi} to reduce variance. In real applications, λ_i should be updated periodically since the rate of clock drifts may vary with time.

Thereafter, the initiator would compensate each node a certain number of *no operations* ($NOPs$) in the processing of their MCUs. Typically, a NOP operation consumes one clock cycle without performing any operation [12]. In our implementation, the MCU runs at a frequency of 4,194,304 Hz. Thus, the granularity of $NOPs$ is about 0.23 μs, indicating that the theoretical MPO would be no larger than 0.23 μs after phase offset compensation. Finally, the number of $NOPs$ is embedded in a dedicated packet sending to each node.

2.3 Limitations on Power Superposition

Since the extreme phase offsets (either too large or too small) are rare, it seems possible to conclude that if we add a transmission with unit amplitude, the amplitude gain of the superposed signal would be $\cos(\frac{\pi}{2} \cdot \frac{0.23\,\mu\text{S}}{T_c}) = 0.75$ with high probability. Hence counting the number of transmissions can be easily done by identifying the power gain of the superposed signal relative to a single signal. Unfortunately, this conclusion is not always true for the following two reasons:

(1) As we will demonstrate in Section 7.1.2, MPO across different transmissions increases with the number of concurrent transmitters. We cannot ensure that MPO is always below a certain value if there are more responders. (2) Constrained by the saturation rate of ADC built in the initiator, sample values exceeding a threshold will be truncated [13], which indicates that there exists an upper bound on the observed received power of a superposed signal.

We conduct an experiment to investigate the observed received power of superposed signals under different responder size. In the experiment, we increase the responder size from 2 to 30 with step length 2. In each round, the poller triggers the selected nodes to perform a transmission simultaneously at maximum output power. Results are shown in Fig. 3, it is observed that the increasing tendency of received power starts to slow down when the responder size exceeds 24. Therefore, the power gain assumption is effective only when the received power is less than a threshold P_0. Clearly,

P_0 depends on the saturation value of the poller's ADC. This threshold is usually lower than the poller's nominal saturation value due to the non-linearity effect of ADCs [13]. Based on the results in Fig. 3, we set $P_0 = 1.2$ mW in our following experiments.

3 System Model

3.1 Single Signal Power Prediction

We consider a static network composed of a poller s and a neighbor set N[1]. Denote by $L(s,i)$ the path loss from a neighbor i ($i \in N$) to the poller s. In this paper, $L(s,i)$ is the effective path loss from i to s that accounts for not only the free space attenuation but also shadowing and multipath effects. Since we target static networks, we suppose path losses change slowly and $L(s,i)$ is relatively constant. Let $P(i,h_i)$ be the received power obtained by s when a single node i responds at power h_i. Similar to the power model in [5], we have $P(i,h_i) = a_i h_i + b_i$, where a_i and b_i are coefficients we need to estimate. Notice that a_i is a scale coefficient of the transmitting power, and b_i depends on the additive background noise $\mathcal{N}$ and the path loss $L(s,i)$: $b_i = \mathcal{N} - L(s,i)$. For a given node, a_i, b_i and $\mathcal{N}$ are all constants.

To predict $P(i,h_i)$, poller s first estimates a_i and b_i on link (i,s). This requires at least two samples of tuple $(P(i,h_i),h_i)$ which can be obtained by either overhearing or directly receiving packets from i. In reality, we use more than 2 of these tuples and estimate a_i and b_i by least-square approximation. After getting the estimation of b_i, denoted as $\hat{b_i}$, we compute the estimation of path loss as $\hat{L}(s,i) = \mathcal{N} - \hat{b_i}$. In the context of this paper, the unit of power is milliwatt while commodity sensor nodes usually provide power with the granularity of 1 dBm. Hence, the poller should first perform a conversion on these two units before the estimation. Finally, power prediction once can be done during the node synchronization phase (see Section 2.2) at no additional cost.

3.2 Power Superposition Model

Here we introduce how to predict the received power of superposed signals. Let I be a non-empty subset of N, $P(I)$ be the received power of the transmissions from nodes in I. By substituting the amplitude A_i in Eq. (1) with $\sqrt{\hat{P}(i,h_i)}$, we obtain the estimation of $P(I)$ as$\hat{P}(I) = (\sqrt{\hat{P}(k,h_k)} + \sum_{i\in I\setminus k}\sqrt{\hat{P}(i,h_i)}\cos(\frac{\pi}{2T_c}\tau_i))^2$ where signal k is the earliest signal reaching s that has zero phase offset. In the implementation, we assume that signal k is from the node who has the least total delay in I after propagation delay compensation. Note that we ignore the noise in signals. However, the background noise $\mathcal{N}$ is not negligible and it is accounted for in $\hat{P}(i,h_i)$.

Now, phase offset τ_i is the only unknown parameter. Nevertheless, it appears rather difficult to estimate such a random parameter. In our experimental results in Section 7.1.2, we show that after signal propagation delay compensation, the distribution of τ_i concentrates to a range from 0.1 to 0.4 μs. For simplicity, we use the average case of 0.25 μs to all delayed signals. Thus, we have

$$\hat{P}(I) = \left[\sqrt{\hat{P}(k,h_k)} + \cos\frac{\pi}{4}\sum_{i\in I\setminus k}\sqrt{\hat{P}(i,h_i)}\right]^2. \quad (2)$$

Substituting all τ_i with 0.25 μs may introduce error in the power prediction. This error and the error introduced in estimating coefficients a_i and b_i (see Section 3.1) are rather difficult to be removed, respectively. We instead examine and mitigate the combined effect of these two kinds of errors. Consequently, we have the following requirement for bounded prediction error by using the above estimation: *there exists some δ, for any non-empty set $I \subseteq N$:* $\hat{P}(I) - \delta \leq P(I) \leq \hat{P}(I) + \delta$. This requirement holds in a static network without interference and our experiments show that the 99th percentile of δ is at most 0.03 mW.

4 Poc Design

Poc allows a poller to estimate the number of neighbors where a predicate holds with constant time. Constrained by the hardware limitations, the received power of superposed signals, however, cannot exceed a threshold P_0. To count the responder size in dense networks, Poc adopts a probabilistic estimation method which will be detailed in this section.

4.1 Overview

Based on the power threshold P_0, we immediately obtain a threshold on the number of simultaneous responders, which is denoted as n_0. In the design of Poc, poller s first broadcasts a response probability p and a predicate Q to all neighbors, each neighbor where Q holds will respond independently with probability p. Thereafter, based on the received power, s can estimate the number of responders. If more than n_0 nodes respond, s may simply discard the result and launch another round of estimation until a satisfactory accuracy has been achieved. Notice that p is a variable in Poc which will be optimized as the estimation process proceeds.

In the rest of this paper, we call the counting result in each round a *sample*. Then the counting problem can be formulated as a parameter estimation problem with constrained samples. Let x_i ($i = 1, \ldots m$) be independent binomial random samples ($B(n,p)$), each with the same number of n of trials and the same probability p of a success on a single trial. If $x_i \leq n_0$ we call it an *effective sample*, otherwise an *ineffective sample*. By effective sample, we mean that its value conveys quantity information. In contrast, ineffective samples only convey a binary state, i.e., whether it is larger than n_0 or not. Our goal is to estimate n with known p based on all samples collected. Note that if $n \leq n_0$, by letting $p = 1$, the poller can directly get $\hat{n}$ without additional operations. Avoiding trivial cases, we only consider the case where $n \in [n_0 + 1,\ N]$ in the following.

4.2 Response Power Assignment

Poc requires that the received power of each node is similar at the poller. In response power assignment, the poller searches for a target received power ξ that all nodes can adjust to within their power level ranges. The poller can compute for each node its transmitting power level to adapt its

1. We abuse the notation N slightly, N in some places also represents the total number of neighbors.

received power to ξ. However, since the set of available power levels is limited (only 8 in CC2420 radios), such ξ's may not always exist. For example, in a topology that nodes have extremely diverse path losses, one node transmitting at the lowest power might still produce larger power than the node transmitting at the highest power. In such cases, Poc doesn't work. However, a typical topology usually doesn't have such diverse path losses. When ξ exists, there might be multiple such ξ's. For simplicity, we choose the one that minimizes the power adjustment gap over all nodes. More mathematically, $\xi = \text{argmin}_{t\in[\alpha,\beta]}(\sum_{i\in N}|P(i,h_i)-t|)$ where α is the received power of the node with least path loss transmitting at the lowest power level, and β is the received power of the node with greatest path loss transmitting at the greatest power level. This computation method is similar to that in [5].

To ensure that different responder size generates sufficiently different received power at the poller, it requires that $\xi > 2\,\delta$. This requirement holds trivially because δ is usually much less than $\frac{\alpha}{2}$. By applying ξ into Eq. (2), we construct the relation between $P(I)$ and the cardinality of I as

$$P(I) = \left[\sqrt{\xi} + \frac{\sqrt{2}}{2}(|I|-1)\sqrt{\xi}\right]^2. \tag{3}$$

Since function $P(I)$ is monotonous, we can get its inverse $g(\cdot)$ and the cardinality of I as $g(P(I))$. Apparently, n_0 can be computed as $g(P_0)$.

4.3 Estimation Methodology

4.3.1 Estimator

According to our previous discussion, the sample space of Poc is $\{0,1,\ldots,n_0\}$. This is smaller than the sample space of standard binomial distribution of $\{0,1,\ldots,n\}$. Since the poller discards all ineffective samples, the probability of observing an effective sample is conditioned on the fact that it is in the smaller sample space $\{0,1,\ldots,n_0\}$. Let p_c be the probability that this condition holds, then the probability mass function could be described as:

$$p_k = \begin{cases} \frac{\binom{n}{k}p^k(1-p)^{n-k}}{p_c}, & 0 \le k \le n_0, \\ 0, & \text{otherwise}, \end{cases} \tag{4}$$

where p_k is the probability that $x_i = k$. The probability that $x_i = n_0$ remains the same because the poller can discriminate the case where a sample is larger than n_0 and the case where a sample is exactly n_0. Clearly, $p_c = \sum_{i=0}^{n_0}\binom{n}{i}p^i(1-p)^{n-i}$. We now compute the expected value and variance of sample X as: $E(X) = \sum_{k=0}^{n_0} kp_k$, $\text{Var}(X) = E(X^2) - E(X)^2 = \sum_{k=0}^{n_0} k^2 p_k - (\sum_{k=0}^{n_0} kp_k)^2$. In Fig. 4, we plot the expected value and variance of X with respect to different n under $p = 0.15, n_0 = 20$. As is shown, $\text{Var}(X)$ is not a monotonous function; it cannot be used as the estimator. On the other hand, $E(X)$ is a monotonous increasing function of n. This is intuitive since under a given p, $E(X) \to n_0$ as n increases. Thus, given a observed value of X, using the inverse function of $E(X)$, we can get the estimation value $\hat{n}$.

To reduce the variance, we conduct multiple rounds of estimation, and use the sample mean $\bar{X} = \frac{1}{m}\sum_{i=1}^{m} x_i$ as our estimator. Here, m is the number of effective samples. Since

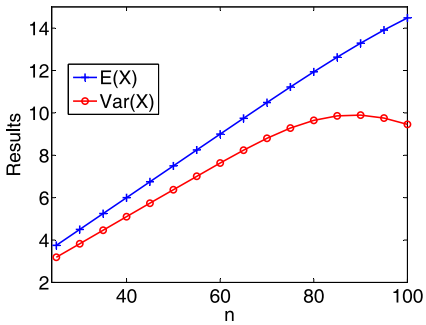


Fig. 4. Expected value and variance of X against different responder size n under $p = 0.15, n_0 = 20$.

$\bar{X}$ is averaged over m independent measurements, its variance becomes $\text{Var}(X)/m$.

4.3.2 Optimize Response Probability p

Consider a requirement that the relative error $\frac{|\hat{n}-n|}{n}$ of our estimator is below a threshold θ with confidence η. In mathematical terms, we request that $P\{|\hat{n}-n| \le \theta n\} \ge \eta$. Let $\mu(n)$ and $\sigma(n)$ be the expected value and variance of $\bar{X}$ under given p and n_0, respectively, $\mu^{-1}(\cdot)$ is the inverse of $\mu(n)$. By substituting $\hat{n}$ with $\mu^{-1}(\bar{X})$, we get

$$\begin{aligned} &P\{(1-\theta)n \le \mu^{-1}(\bar{X}) \le (1+\theta)n\} \\ &\quad = P\{\mu((1-\theta)n) \le \bar{X} \le \mu((1+\theta)n)\} \ge \eta. \end{aligned} \tag{5}$$

Let $Y = \frac{\bar{X}-\mu(n)}{\sqrt{\sigma(n)}}$, then (5) can be rewritten as

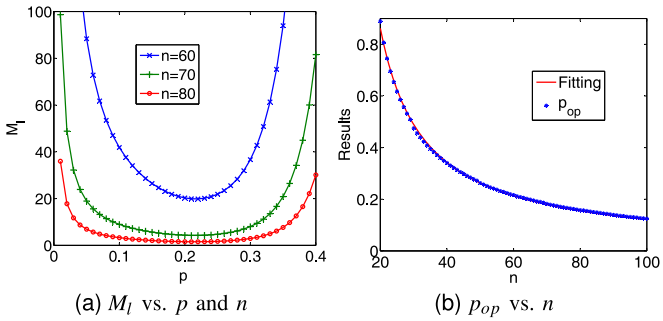
$$P\left\{\frac{\mu((1-\theta)n)-\mu(n)}{\sqrt{\sigma(n)}} \le Y \le \frac{\mu((1+\theta)n)-\mu(n)}{\sqrt{\sigma(n)}}\right\} \ge \eta. \tag{6}$$

We know that Y approximates a standard normal random variable based on the central limit theorem [14]. Denote the lower bound and upper bound of Y by y_l and y_u respectively, i.e., $y_l = \frac{\mu((1-\theta)n)-\mu(n)}{\sqrt{\sigma(n)}}$, $y_u = \frac{\mu((1+\theta)n)-\mu(n)}{\sqrt{\sigma(n)}}$. We numerically found that $y_l \approx -y_u$ always holds for $\theta \le 0.1$, $n \le 200$. Thus, it is safe to assume that $y_l = -y_u$, indicating that the confidence interval is symmetric on both the upper and lower sides of n. Applying this symmetry property and $\sigma(n) = \text{Var}(X)/m$ into (6) and rearranging for m yields

$$m \ge \frac{\text{Var}(X)[\Phi^{-1}(\frac{1+\eta}{2})]^2}{(\mu((1+\theta)n)-\mu(n))^2}, \tag{7}$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution. Inequality (7) gives a lower bound on the number of effective samples that is required to satisfy a desired relative error θ with confidence η. Since m follows binomial distribution $B(M,p_c)$ where M denotes the total number of samples. Let m_{min} be the right hand side of (7), then the lower bound of the expected value of M is

$$M_l = \frac{m_{min}}{p_c} = \frac{\text{Var}(X)[\Phi^{-1}(\frac{1+\eta}{2})]^2}{(\mu((1+\theta)n)-\mu(n))^2 p_c}. \tag{8}$$



(a) M_l vs. p and n (b) p_{op} vs. n

Fig. 5. Convexity of M_l with respect to p and the corresponding p_{op} for different number of n. Results are obtained under $\theta = 0.05$, $\eta = 0.95$.

Note that M_l is the estimation delay, and our goal is to find an optimal response probability p_{op} in order to minimize M_l. In Fig. 5a, we show that M_l is a convex function with respect to p under a given n, thus such an optimal value p_{op} always exists. In Fig. 5b, we compute p_{op} numerically with respect to different n under $\theta = 0.05$, $\eta = 0.95$. The results demonstrate that p_{op} decreases monotonically with an increasing n. Using the Matlab curve fitting tool, we obtain the relation between p_{op} and n as $p_{op}(n) = 11.35/(n-7)$.

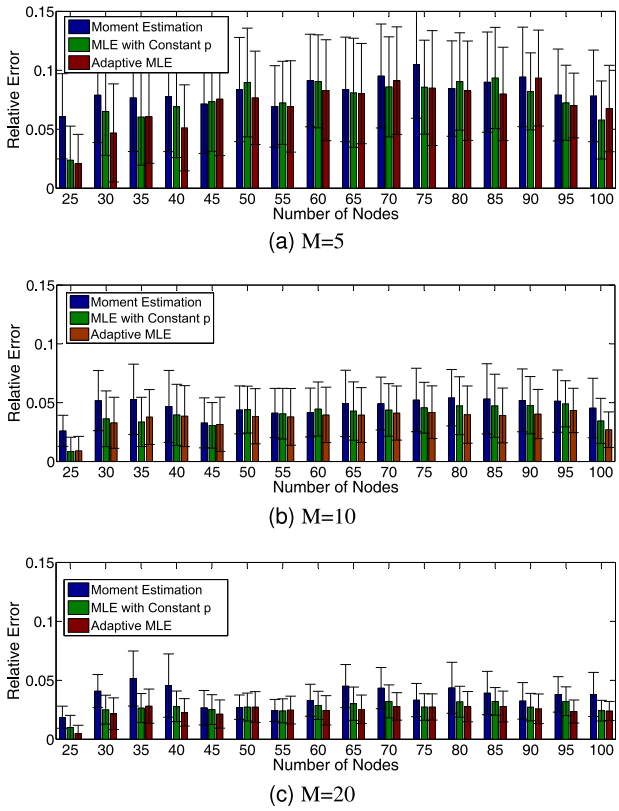
4.3.3 Adaptive Estimation

As shown in Fig. 5b, in order to find p_{op}, we must know n in advance. However, n is the number we aim to estimate, which is unknown at the initial stage. In this section, we propose an adaptive estimation method to estimate n: the poller first performs an estimation $\hat{n}$ with a randomly selected p over range $[n_0+1, N]$; in the second step, the poller use $\hat{n}$ to update the response probability and launch another round of estimation. In each round, all the samples collected in previous rounds are used to calculate a new $\hat{n}$. Finally, the poller could repeat the process until either $\hat{n}$ has little variability or it has reached a certain number of rounds. In our implementation, the poller stops when M rounds has been reached.

Our former analysis only considers effective samples. An ineffective sample, however, also contains useful information about the parameter n. In each step of the adaptive estimation, we take advantages of both the effective and ineffective samples and use the maximum likelihood estimation (MLE) to give a relatively accurate estimation of n. For example, assume that we obtain M samples among which a samples are ineffective and b samples are effective. Effective samples are denoted as $x_1, x_2, \ldots x_b$, and the probabilities of getting these samples are $p_1, p_2, \ldots p_b$, respectively. Ineffective samples are obtained with the condition probabilities p_c's being $p_c^1, p_c^2, \ldots, p_c^a$. The likelihood function $L(n)$ is

$$L(n) = \prod_{i=1}^{a}(1-p_c^i) \cdot \prod_{i=1}^{b}\left(\binom{n}{x_i} p_i^{x_i}(1-p_i)^{n-x_i}\right). \quad (9)$$

Then, $\hat{n}$ is returned as the argument over $[n_0+1, N]$ that maximizes $L(n)$. Finally, we summarize the whole procedure of adaptive estimation in Algorithm 1.



(a) M=5 (b) M=10 (c) M=20

Fig. 6. Estimation error under different M.

Algorithm 1. Adaptive Estimation

Input: n_0, N, θ, η, M
Output: $\hat{n}$

Get received power sample P_r using $p = 1$;
if $P_r \leq P_0$ **then**
  return $g(P_r)$;
$i = 1$; $S = \emptyset$; $P = \emptyset$;
p = random number between $p_{op}(n_0+1)$ and $p_{op}(N)$;
n_{prev} = random number between n_0 and N;
  while $i \leq M$ **do**
  s = sample in a round with p;
  $S.add(s)$; $P.add(p)$;
  $\hat{n} = MLE(n_0, N, S, P)$;
  $p = p_{op}(\hat{n})$ // adaptively update p
  $n_{prev} = \hat{n}$;
  $i = i + 1$
return n_{prev};

4.3.4 Evaluation of Adaptive Estimation

To evaluate the performance of adaptive estimation, we design two baseline estimation methods. In the first baseline, we use the first-order moment estimation method which discards all ineffective samples and compute $\hat{n}$ as $\hat{n} = average(\textit{effective samples})/p$. In the second baseline, we use all samples and use the MLE method to compute $\hat{n}$ but keep p unchanged. Fig. 6 shows the results with $M = 5$, $M = 10$, and $M = 20$, respectively. Each data point is average over 1,000 runs. It is observed that the two MLE's with constant p and adaptive p are both notably better than the moment estimation. This is because the moment estimation doesn't exploit the information conveyed in ineffective

samples. While the two MLE methods are very close in accuracy, the adaptive estimation is slightly better, especially when M is large than 10. Adaptive estimation's advantage is due to the fact that updating p in each round is more likely to find an optimal p with which a sample conveys more information than with a less optimal one. The average accuracy of Adaptive Estimation is 7.6, 3.4, 2.8 percent respectively, indicating the fact that larger M deliver higher accuracy. However, $M = 10$ is large enough in our system as the accuracy gain obtained by increasing M from 10 to 20 is less than 1 percent.

4.3.5 A Possible Extension of Adaptive Estimation

A possible improvement over adaptive estimation is that nodes who have responded in a slot keep silent in all subsequent slots. This can be implemented as follows. In slot 1, the poller broadcasts a random p and get x_1 counts. In slot 2, the poller estimates n and broadcasts the optimal probability for $\hat{n} - x_1$ and get x_2 counts, and so on. Meanwhile, if the poller's estimation of n is no larger than n_0, then it will set $p = 1$ in the next slot and get counts x_t. If $x_t \leq n_0$, the poller will stop and estimate exactly the number of nodes $n = \sum_{i=1}^{t} x_i$. The poller also stops if the total number of slots M has been reached. In the process, if the counts in some slot is ineffective, the poller can simply discard the result and go to the next slot.

The extended version of adaptive estimation may have three important advantages. (1) Energy efficiency: nodes don't need to respond in every slot. (2) Less congestion: the number of responders is decreasing as the estimation process goes. (3) High accuracy: higher probability of getting an effective sample. Due to the space limitation, we delegate the experimental study of this extended version to our future works.

5 Poid Design

Beyond fast and fine-grained counting, we show in this section that the power superposition model can also be used to achieve accurate identification with constant delay.

5.1 Overview

The identification problem can be formulated as follows: *A poller s broadcasts a predicate Q. Let Ω be the set of neighbors where Q holds ($\Omega \subseteq N$). Neighbors in Ω that have received the predicate transmit an identical ACK simultaneously. The poller overhears the concurrent transmissions and checks the identities of all the nodes in Ω based on the received power.*

The workflow of Poid contains three steps: (1) assign response power; (2) broadcast Q and measure the received power; and (3) identify the nodes that have responded. Notice that the first step is done offline for one-time only, and the second and the third steps are two routines that can be finished with constant delay.

5.2 Response Power Assignment

Unlike Poc, nodes in Poid are assigned diverse power levels such that any non-empty set of responders will generate unique received power at the poller. We propose a simple yet efficient algorithm in Algorithm 2 to achieve this goal. In Algorithm 2, we compute an assignment $\{h_i\}$ for each node $i \in N$ using a simple rule: the larger the path loss of a node i, the lower the assigned power. In this way, we ensure that the received power is sufficiently separated. Line 4 assigns the minimum power to node 1 such that $P(1, h_1) - \delta$ is above P_l, the weakest response power that can be detected by the poller. For each of the remaining nodes, it searches for a minimum power level such that the received power is at least $2\,\delta$ larger than the mix power of all nodes in I. Notice that when calculate the mix power, Algorithm 2 neglects the phase offset of all nodes' signals (line 8). In the following theorem, we prove that Algorithm 2 guarantees identifiability.

Algorithm 2. Response Power Assignment

Input: Path loss of each neighbor $L(s, i)$ Available power level set H
Output: Transmitting power h_i for each node i
1 Label the neighbors according to their path losses such that $L(s, i) \geq L(s, i + 1)$;
2 Label the elements in H such that $H_k < H_{k+1}$;
3 $I = \emptyset$;
4 $h_1 = argmin_{h \in H}(P(1, h) - P_l \geq \delta)$;
5 $I = I \bigcup$ node 1 ;
6 **for** $i = 2$ to N **do**
7 **for** $k = 1$ to $|H|$ **do**
8 **if** $P(i, H_k) - [\sum_{i \in I} \sqrt{P(i, h_i)}]^2 \geq 2\,\delta$ **then**
9 $h_i = H_k$;
10 $I = I \bigcup$ node i;
11 **break**;

Theorem 1. *For any two given subsets S_i and S_j of node set N, if $S_i \neq S_j$ then $P(S_i) \neq P(S_j)$.*

Proof. See Appendix A in the supplemental file, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TMC.2015.2416173. □

5.3 Finger-Print Based Identification

After obtaining the received power P_r, the poller maps P_r to a certain set of neighbors. The mapping method is quite intuitive: find a subset $\hat{I}$ of N that minimize $|P(\hat{I}) - P_r|$, and return $\hat{I}$ as the result. To further reduce the complexity, we pre-estimate the expected received power of any possible sets of neighbors and construct a finger-print table that maps each received power level to a set of neighbors. Let $\mathscr{G}$ be the set of all possible (expected) received power levels that could be experienced by s, $\forall g \in \mathscr{G}$, $L(g)$ denote the list of nodes that will generate g at the poller. Then, s returns the set of responders $\hat{I}$ satisfying $\hat{I} = L(\text{argmin}_{g \in \mathscr{G}}\{|P_r - g|\})$.

5.4 Constraints and Extensions

Due to the hardware limitations presented in Section 2.3, Poid only applies to sparse networks with relatively small neighborhood size. We note that sparse networks are often encountered. Examples include forest fire monitoring [15], permafrost monitoring [16] and battlefield monitoring [17]. In these scenarios, a sink or a mobile agent in the case of Data MULEs [18] usually has small

neighborhood size, thus can benefit from Poid to achieve efficient node identification.

In dense networks, we can extend Poid as follows: since we can count the number of responders in a single time slot, we can let the poller listen for multiple time slots after broadcasting the query. Each node would choose a certain set of slots to respond. Assuming that the indices of slots chosen by each node are known as priors, then the poller can identify the responders by counting the responses in each slot. We term this approach counting-based identification (CBI).

A major concern for CBI is: how long will the poller listen? i.e. what is the number of time slots required by the poller to perform identification? If the number is near to the network size n, then it will be no better than the TDMA-based method. The following theorem bound the number of time slots required by CBI:

Theorem 2. *Let m be the total number of time slots required by CBI, if there is no limit on the number of responses in a single slot, then m satisfies $n/\log(n+1) \leq m \leq n - \log(n/\log(n+1) - 1)$.*

Proof. See Appendix B in the online supplemental file. □

We note that the upper bound of time slots is still $\Theta(n)$, which is not a significant improvement over TDMA. Therefore, the application of Poid in dense networks is very restricted to those delay-critical applications (e.g., [19], [20]). Poid is also constrained by the hardware limitation: the poller cannot count too many responses in a single slot. Fortunately, we can use the same probabilistic estimation method used by Poc: let each node respond with a probability p in the slots it has chosen, then the poller estimate the number of responses in each slot and identify the responders. Obviously, in this scheme, the ultimate identification accuracy is significantly impacted by the counting accuracy in each single slot. Nevertheless, systematic experimental studies on the impact of counting on identification are beyond the scope of this paper, we leave it to our future works.

6 Implementation

We implement Poc and Poid on a testbed consisting of 1 GNURadio/USRP and 50 TelosB nodes. The USRP runs the UCLA ZigBee PHY [21] code to send 802.15.4 packets. We set the sampling frequency of USRP to be 2 MHz, which is equivalent to that of the CC2420 radios [22]. Both USRP and TelosBs work on ZigBee Channel 26 which experiences little WiFi interference.

6.1 Received Power Measurement

Raw samples obtained by USRP are just complex numbers with the real part being the I-phase component and the image part being the Q-phase component. The poller s computes the sequence of $I^2 + Q^2$ as power samples. Recall that the default length of an ACK response is 352 μs, the number of power samples that s needs to record is 2 MHz·352 μs = 704. Denote the sequence of power samples by S, then s returns the median of 10 peaks of S as the received power. Meanwhile, we empirically measured that the background noise is about 0.009 mW, the power from a node that has the largest path loss and transmits at the lowest power level is 0.04 mW (i.e., $P_l = 0.04$ mW). Thus, once s sense a energy elevation that is above 0.04 mW, it starts recording the power samples.

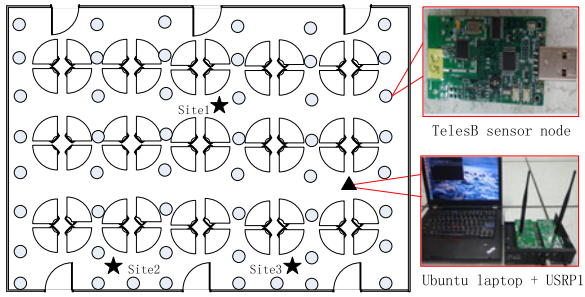


Fig. 7. Physical layout of 50 nodes and a USRP in a 18×25 m laboratory.

6.2 Handling External Interference

We consider three cases where interference can occur and introduce corresponding handling mechanisms.

False positive energy elevation. In this case, the energy elevation sensed by s is caused by interference rather than the transmissions of responders. We exploit the fact that the time gap t_Δ between the instant the last bit of Q is transmitted and the instant the first bit of an ACK is transmitted is known (in CC2420 radios, $t_\Delta \approx 192\,\mu$s) [22]. Let ϵ be the error tolerance[2] of t_Δ. The poller s will start listening immediately after the last bit of Q is transmitted and ignore any energy elevation whose delay is either smaller than $t_\Delta - \epsilon$ or larger than $t_\Delta + \epsilon$.

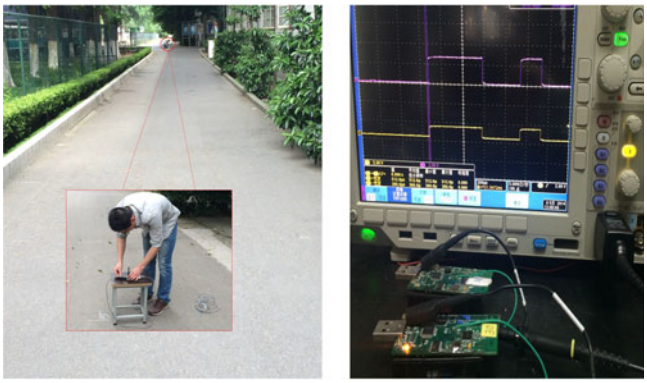
Partial peaks pollutions. In this case, several peaks of S are polluted by random impulses or background noises. Since s discards all non-peak power samples, we only consider the case that the polluted peaks are larger than the normal. We first select 100 peaks of S and compute their mean μ and variance σ, and then discard all the peaks above $\mu + 3\sigma$. Thereafter, we randomly choose 10 peaks from the remnant and return the median of them as the received power.

Overlapped with other ZigBee transmission. In this case, the transmissions of the responders are partially or entirely covered by a ZigBee transmission in neighboring regions. To detect this external interference, the poller s exploits the fact that the duration of an ACK is known as a priori and can reject a measurement if unexpected response length is observed.

7 Experimental Results

In this section, we evaluate the performance of Poc and Poid under different wireless environments and parameter settings. The testbed is located in a 18×25 m laboratory of a school building. As illustrated in Fig. 7, 50 sensor nodes are placed in a grid-like topology. The USRP is connected to a Ubuntu 12.04 laptop and placed at the right area of the lab. In the initial stage, we let the poller perform 30 packet exchanges with each node. The response packet of each node contains their respective transmitting power and the instants of transitions on their SFD pins. Based on the information embedded in

2. We empirically set ϵ to be a symbol's duration, i.e., 16 μs.



(a) $d = 105\,m$ (b) $d = 0\,m$

Fig. 8. Measuring signal propagation delay with varying distance d. When $d = 0$, both of the SFD pins of the poller and the receiver are connected to the oscilloscope for reference.

the response packets and the observed received power, the poller s will (1) compensate signal propagation delay for each node; (2) estimate the channel coefficients of each node; (3) compute the path losses of each node, and (4) complete the power assignment for each node.

7.1 Micro-Benchmark

We conduct micro-benchmark to (1) investigate the relationship between signal propagation delay (SPD) and distances; (2) measure the interference impact in our testbed; (3) measure the distribution of power prediction error δ, and (4) evaluate Poc's performance under controlled responder size.

7.1.1 Measuring and Compensating Signal Propagation Delay

Signal propagation delay is signal's two-way around flight time between the poller and a receiver. This delay mainly depends on the distance between them. We use two TelosB nodes, one acts as the poller and the other acts as a receiver. To minimize the uncertainty delay brought by software processing, the receiver is programmed to immediately schedule an ACK transmission after the rising edge of it's SFD pin. The SFD pin of the poller is connected to an oscilloscope with a granularity of 2 ns. We conduct the experiment on a campus sidewalk, which is shown in Fig. 8.

SPD is computed by subtracting the receiver's hardware delay and software delay from the duration of two falling edges of the poller's SFD pin (i.e., $t_9 - t_4$ in Fig. 2). To measure the hardware delay and software delay of the receiver, we also connect the receiver's SFD pin to the oscilloscope, and the summation of these delays is computed as $t_s + t_h + t_{pr} + t_f$ showing in Fig. 2. Note that SPD in our paper also include radio's processing delay and it can be roughly computed by halving the measured SPD when the poller and the receiver are placed together, as is shown in Fig. 8b.

Results are plotted in Fig. 9. We note that every 15 meter's distance increase would add approximately $0.1\,\mu$s to the measured SPD, this is in consistent with our knowledge that $(15 \times 2)/c = 0.1\,\mu$s, where c is the speed of light. It can also be seen that a distance gap of 120 m causes more than

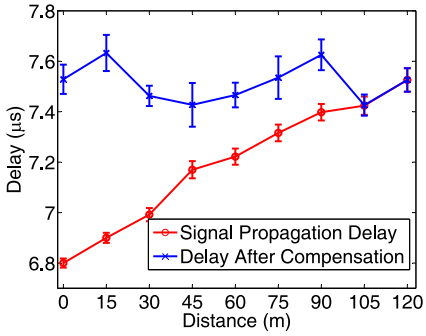


Fig. 9. Signal propagation delay against the distance and the synchronization level after compensating this delay.

0.7 μs on the phase offset of two signals, which even exceeds the threshold of CI, indicating that SPD must be compensated before applying either Poc or Poid in real applications. We then insert several NOP instructions to the receiver to make it's SPD similar to the largest one. The number of instructions for the receiver of each distance is 3, 3, 2, 1, 1, 1, 1, 0, 0, respectively. Then, we re-conduct the experiment after inserting instructions. Due to the 0.238 μs granularity of a NOP instruction, we see from the blue line that the delay differences across different distances vary right within this variability. Also, we observe that delay variances increase after inserting instructions. This may be due to the inherent time uncertainty of software processing.

7.1.2 Synchronization Precision Achieved

We randomly selected 31 TelosB nodes in our testbed shown in Section 7. One of them acts as the poller and the remainder are responders. The poller's SFD pin is connected to an oscilloscope. In the initial stage, we let the poller perform 30 packet exchanges with each neighbor to compensate their respective propagation delays. Then the experiment proceeds in rounds. In each round, the poller randomly chooses k neighbors and records its t_c^0 every time after a packet exchange with each neighbor. We know that if neighbors are perfectly synchronized, t_c^0 will always be the same. Therefore, by computing the maximum difference among different t_c^0's, we can measure the MPO among k neighbors' transmissions. In each round, we repeat the experiment for 200 times.

Fig. 10 compares the synchronization accuracy of Glossy [6] and the one after adding the propagation delay compensation (PDC). Dots on the lines show the average phase offsets; error bars indicate the standard deviation. The results

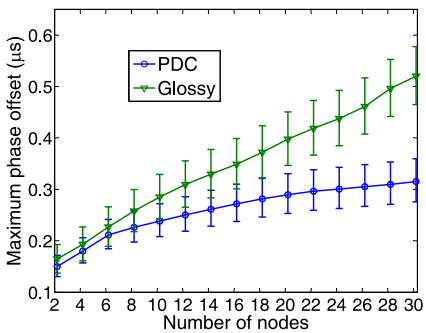


Fig. 10. Synchronization error against different number of simultaneous transmissions.

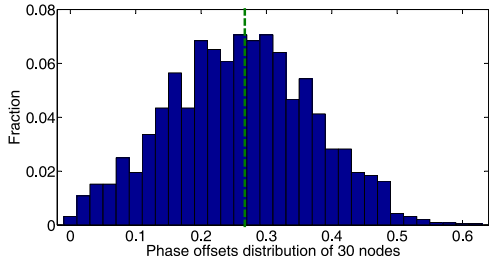


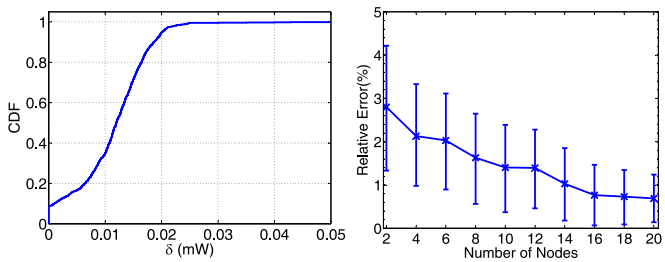
Fig. 11. Distribution of phase offsets among 30 nodes. Results are obtained by computing the difference of each t_c^0 relative to the smallest one. Mean value (dotted line) is 0.268 μs.

of Glossy are obtained by applying its synchronization methods (i.e. methods of software delay compensation and hardware delay compensation) to our testbed. We observe that as the number of nodes increases, the synchronization accuracy of Glossy degrades much faster than PDC and even exceeds $0.5\,\mu$s with more than 28 nodes. Although the MPO of PDC increases notably with the increasing number of nodes, its maximum values are always less than $0.4\,\mu$s.

Fig. 11 further gives the degree of synchronization precision we are able to achieve when the responder size is 30. The results are obtained through 20 rounds' experiments. In each round, we change the topology of the 30 responding nodes (as re-selecting the 30 responders in the testbed shown in Section 7). We compute the difference of each t_c^0 relative to the smallest one so as to record the synchronization error of 30 responders. The 90th percentile of the phase offsets is 0.4 μs and the mean offset is 0.268 μs, somewhat worse than the 0.23 μs of theoretical minimum phase offset. However, this level of precision is adequate to support Poc and Poid, which we will demonstrate in the following.

7.1.3 Distribution of Power Prediction Error δ

Next, we measure the distribution of δ. Twenty nodes which are closest to the poller are selected in this experiment. The number of responders monotonically increases from one to twenty. In each round, the poller will first assign each node a transmitting power such that all nodes have similar received power at the poller. Then upon receiving a broadcast packet from the poller, the responders transmit ACKs simultaneously with assigned power level. Prediction error δ is computed as the gap between the expected power and the observed power. We repeat each round for 200 times. The cumulative distribution function (CDF) of δ is plotted in Fig. 12a. We see that δ has 99th percentile being at most 0.03 mW. Thus, the requirement of bounded prediction error is satisfied.



(a) Power prediction error (b) Counting accuracy

Fig. 12. Evaluation of the power superposition model.

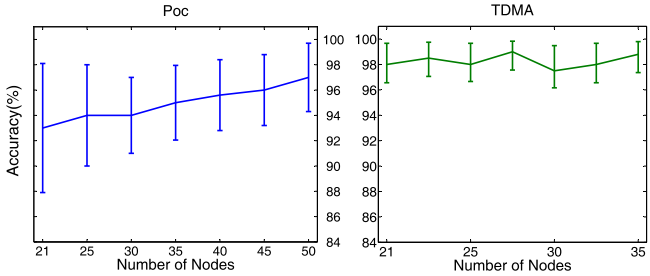


Fig. 13. Counting performance of Poc and TDMA.

7.1.4 Counting with Controlled Responders

We then use $\delta = 0.03\,\text{mW}$ to evaluate the performance of Poc under limited responder size. Experiments are conducted under the same setting as in Section 7.1.3. Based on Eq. (3), we get $n_0 = 20$ here. Results are plotted in Fig. 12b, averaged over 200 estimations for each responder size. When there is no interference, the relative error is as low as 1.5 percent and is nearly always below 4 percent. Note that the error show a decreasing tendency when the number of nodes increases. This is due to the decrease of power variation when the responder size increases, which is also observed in Fig. 3.

7.2 Performance of Poc and Poid

For Poc, we evaluate it's counting accuracy with 10 samples (i.e., $M = 10$). For Poid, we evaluate both the counting accuracy and identification accuracy with only one sample.

Poc. To the best of our knowledge, there is no fast and fine-grained neighbor counting methods in dense WSNs till now. For ease of comparison, we implemented a baseline according to the TDMA scheme described in [4]. We choose not to compare Poc with LogPoll since LogPoll counts nodes on a logarithmic scale, that is to say, if the responder size falls into the range between two consecutive numbers that are integer power of 2, the average error may be very high. In the baseline, each node is assigned a unique time slot. After broadcasting a predicate, the poller can then identify the number of responders based on the number of slots in which the energy power is above the noise floor.

We conduct 200 rounds of experiments for each number of responders varying from 21 to 50. Counting accuracy is shown in Fig. 13. In the experiments, TDMA still cannot count 100 percent accurately because it is also impacted by sporadic external interference. We observe that both the variance and average error of TDMA are lower than that of Poc. However, the performance of Poc becomes better with n increases. For $n \geq 30$, the average accuracy gap between Poc and TDMA is less than 2.5 percent. On the other hand, TDMA uses a number of slots that is linear to the network size while Poc uses only a constant number of slots. Note that in this paper, we refer to the energy consumption as the radio's on-time, particularly, the radio on-time of the poller, thus we believe that Poc is better than TDMA for its significant energy efficiency advantage.

Poid. Due to the hardware limitation and the exponential space size of the finger-print table used in Poid, we use only nine nodes. We note that nine nodes are a reasonable neighborhood size in sparse networks. The responder set changes

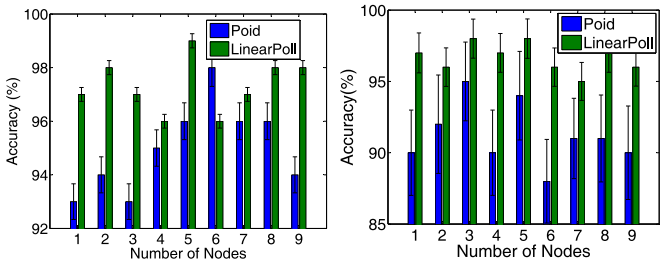
TABLE 1
Interference Caused by a Jammer under Different Settings

Output Power \ Packet Rate	$k=5$		$k=10$		$k=20$		$k=40$	
	P (mW)	F (percent)	P (mW)	F (percent)	P (mW)	F (percent),	P (mW)	F (percent)
−10 dBm	0.07	0.5	0.07	1.1	0.07	2.1	0.07	4.5
−5 dBm	0.12	0.6	0.13	1.1	0.12	2.1	0.12	4.5
0 dBm	0.21	0.5	0.21	1.1	0.20	2.0	0.21	4.5

from round to round and all nodes in the testbed are covered. We compare Poid with LinearPoll [5], a power based identification method also used in sparse networks. In LinearPoll, each neighbor is assigned a unique response length as well as a unique response RSSI. The poller identifies nodes based on the RSSI drops in a receiving procedure. We set the response length difference $\Lambda = 4$ in LinearPoll, which reports the best performance.

We use Ω to denote the set of responders, and then define counting error $e_{|\Omega|}$ and identification error e_Ω as: $e_{|\Omega|} = \frac{|\hat{\Omega}|-|\Omega|}{|\Omega|}, e_\Omega = \frac{|\hat{\Omega}\backslash\Omega|+|\Omega\backslash\hat{\Omega}|}{|\Omega|}$. Results are demonstrated in Fig. 14. For LinearPoll, the average counting accuracy and average identification accuracy is 97.2 and 96.9 percent, respectively. For Poid, the average counting accuracy and average identification accuracy is 95.1 and 91 percent, respectively. While the counting performance is similar, Poid's identification accuracy is notably lower than that of LinearPoll. This is because Poid soley rely on the received power to identify responding nodes; a slight error occurred in the counting results may cause a very different set of responders to be identified. Also, LinearPoll is relatively more stable than Poid when we compare the accuracy variance of them.

However, LinearPoll uses both response length and received power to identify nodes, and all nodes in Poid have identical minimum response length. As a result, by avoiding assigning different response length to different nodes, Poid shows significant less communication delay compared with LinearPoll. We then measure the poller's radio on-time for Poid and LinearPoll respectively in Fig. 15, each result is averaged over 200 rounds. Since in LinearPoll, the response lengths must be four RSSI samples (i.e., 32 symbols) apart, adding a new node to the responder set will cause the radio to listen at leat 32 symbols' duration (0.512 ms) longer. Thus, the radio on-time of LinearPoll increases linearly with the number of nodes. On the contrary, Poid's radio on-time remains constant and is always below 1 ms.



(a) Counting Accuracy $(1 - e_{|\Omega|})$ (b) Identification Accuracy $(1 - e_\Omega)$

Fig. 14. Performance comparison between Poid and LinearPoll.

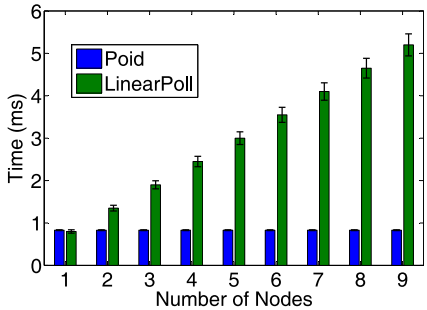


Fig. 15. Poller's radio on-time against different number of responders.

7.3 Impact of External Interference

The setting here is the same as that in Section 7.2, except that we placed an additional node in the testbed. This node acts as a jammer that broadcasts jamming messages at a rate of k packets/s. In the following experiments, we vary k to examine the impact of the frequency of interference. The jammer at the location of site 2 marked in Fig. 7; about 16 m away from the poller. The output power of the jammer is chosen from −10, −5 and 0 dBm[3]. The packet length of the jammer is fixed as 30 bytes, which is 9 bytes larger than that of an ACK of the normal nodes.

7.3.1 *Interference Type*

To better understand the interference level at the poller, we let the poller listen on channel 26 for 30 seconds for each setting of the jammer. The average received power intensity (denoted as P) and the total duration percentage of the interference in 30 seconds (denoted as F) are compared and summarized in Table 1. Since the total duration of interference is proportional to the packet rate given that the packet length is fixed, we see the percentage of interference duration is approximately doubled when k is doubled. The received power[4] is similar and varies little with packet rate if the output power is fixed. Therefore, in our following experiments, we mainly consider four types of interference.

- Interference A: output power is −10 dBm and packet rate is 5 pkts/s;
- Interference B: output power is −5 dBm and packet rate is 10 pkts/s;
- Interference C: output power is 0 dBm and packet rate is 20 pkts/s.
- Interference D: output power is 0 dBm and packet rate is 40 pkts/s

3. In CC2420 radios, 0 dBm is the maximum output power.
4. The power is calculated by USRP and might not be proportional to that in TelsoB platforms due to different antenna parameters.

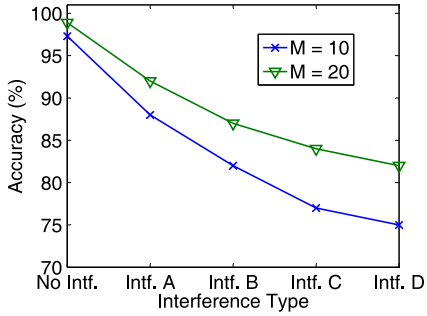


Fig. 16. Counting performance of Poc under different interference types.

Obviously, the severity of Interference A, B, C and D is increasing step by step. Note that the rate of 40 pkts/s is not often occurred in real WSNs since at this rate the battery of a sensor node drains very quickly. We use this high frequency of interference to test the point where our system fails to deliver a good performance.

7.3.2 Performance of Poc under Interference

In this experiment, the responder size is fixed as 50 and the performance of Poc is tested under different types of interference. In each type of interference, the poller collects M samples and use the adaptive MLE method to estimate the responder size. The results with $M = 10$ and $M = 20$ are compared and demonstrated in Fig. 16. Each data point is averaged over 100 runs. Apparently, the performance of Poc degrades notably under our artificially created interference types. Compared with the case with no interference, the accuracy of degradations are 9.3, 15.5, 20.5 and 23 percent for Interference A, B, C, and D, respectively. Especially, under Interference D, Poc's accuracy is only 75%. On the other hand, we see that increasing the number of samples collected by the poller mitigates the sharp performance degradation of Poc. Particulary, the accuracy still remains above 85 percent in the most unpleasant environment. Therefore, while Poc might not be effective in scenarios with such severe external interference, increasing M is a possible mitigation strategy.

7.3.3 Performance of Poid under Interference

The settings in this experiment is the same as that in Section 7.3.2, except that the responder size is changed to be 9. Fig. 17 shows the identification accuracy of Poid. We see that the performance of Poid is also unsatisfactory under extremely unamicable scenarios. Specifically, Poid can only

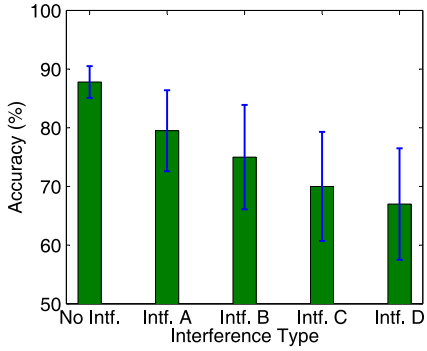
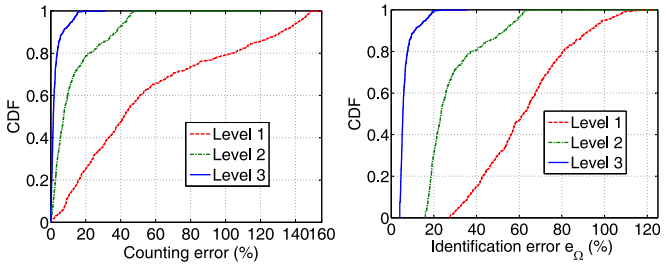


Fig. 17. Identification performance of Poid under different interference types.



(a) Counting error for Poc (b) Identification error for Poid

Fig. 18. Impact of synchronization on the performance of Poc and Poid.

identify about 68 percent responders under Interference D. Therefore, it is suggested that one shouldn't apply our current design of Poid only in environments with heavy interference. For dense networks with a relatively unpleasant wireless environment, the multi-slots method of Poid discussed in Section 5.4 might be a solution. We leave the systematic and experimental study of this method to the future work .

7.4 Impact of Synchronization

As POs of component signals heavily depend on the synchronization accuracy among simultaneously transmitters. We investigate the performance of Poc and Poid under three different synchronization accuracy levels.

Level 1. No delay is mitigated or compensated. Responders are just synchronized by a common trigger.

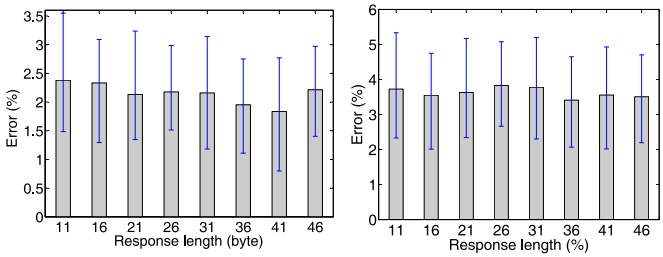
Level 2. Software delay is compensated and hardware delay is mitigated. This synchronization level is used by Glossy [6].

Level 3. In addition to level 2, level 3 also evaluates and compensates the round-way signal propagation delay from each responder to the poller. This synchronization level is used in the previous experiments.

The experiments are conducted on channel 26. For each synchronization level, we collect 1,000 data trace for both Poc and Poid. For simplicity, only identification accuracy is considered for Poid. The number of samples M in Poc is set to 15. Fig. 18a plots the CDF of relative error of Poc under different synchronization levels. We observe that both of levels 2 and 3 show significant accuracy advantage over level 1. For 80th percentile, the relative error is 5, 20 and 100 percent for level 3, 2 and 1 respectively. For counting error's CDF of up to 20 percent, level 1 sees only 25 percent while level 2 sees 80 percent and level 3 sees 100 percent. We also observe similar results for Poid in Fig. 18b. Therefore, Fig. 18 demonstrates that synchronization accuracy among responders plays a key role in the performance of our systems. In our testbed, the maximum distance difference from neighbors to the poller is about 28 m, thus the maximum gap among round-way signal propagation delays would be at least 0.187 μs. By reducing the phase offsets caused by this delay, level 3 shows a 10 percent decrease on the average error of Poc and 21 percent decrease on the average error of Poid when compared with Glossy.

7.5 Impact of ACK Length

In the former experiments, we use the default ACK length of 11 bytes. We now change the length of ACKs and explore the performance of Poc and Poid under different response



(a) Counting error for Poc (b) Identification error for Poid

Fig. 19. Impact of packet length on the performance of Poc and Poid.

lengths. We re-conduct the experiments with eight different ACK lengths and the settings are the same as in Section 7.4. Average error of Poc and Poid, plotted in Fig. 19, are both fairly constant and very similar (within 0.5 percent variation) to the results of the shortest response length (11 bytes), thus showing no significant dependency on the ACK length. This may due to the fact that ACK length has little impact on the received power of the poller. Since shorter response length implies shorter communication delay and higher energy efficiency, Poc and Poid should always assign the shortest response length to all neighbors.

8 Related Works

Numerous existing studies addressed the issue of efficient responses collection in wireless networks. Basically, these works can be categorized into multi-carrier based and time slots based. Works based on multi-carriers exploit the orthogonal frequency division multiplexing (OFDM) modulation scheme to let nodes respond in different sub-carriers simultaneously [23], [24]. Although this scheme can achieve counting and identification in constant delay, it is not suitable for sensors for its heavy computation burden brought by Fourier Transformations. Time slots based methods have been extensively studied in the field of counting RFID tags [25], [26], [27], [28]. A common feature of these methods is that the poller can only distinguish three different events per given slot: empty slot (no transmission in this slot); singleton slot (one node transmit in this slot) and colliding slot (two or more nodes transmit in this slot). Since Poc and Poid can further distinguish the number of nodes when multiple nodes transmit in a same slot, we believe that integrating our methods into the existing RFID estimation schemes will significantly improve counting accuracy and reduce counting delay.

In their seminal work of [5], Zeng et al. proposed RSSI-based counting schemes LinearPoll and LogPoll, which allow nodes respond simultaneously on a single carrier. LinearPoll, worked in sparse networks, identify nodes by their response length as well as RSSI and consumes energy that is linear in the neighborhood size. LogPoll works in dense networks and counts nodes on a logarithmic scale and consumes constant energy. Compared with their works, we utilize power information with higher resolution and demonstrate two substantial advantages over LinearPoll and LogPoll. Specifically, Poid reduces the energy cost of LinearPoll from $O(n)$ to $O(1)$ while Poc improves the counting accuracy of LogPoll from logarithmic scale to linear scale.

Constructive interference is an emerging trend in wireless communications. It allows a common receiver to decode concurrent transmissions of an identical packet and has been exploited to achieve fast network flooding [6], [29], enhancement of overall link PRR [11], [30] and fast data dissemination [7]. A common requirement of these works is that the superposed signals are decodable at the receiver. Instead, Poc and Poid take advantage of the received power of superposed signals and do not need to decode them, supposed to be more energy efficient than the decoding-based methods.

9 Conclusions

This paper investigates the problem of simultaneous query via constructive interference in WSNs. We present Poc and Poid. Poc allows neighbors to respond to a common poller simultaneously and estimate the number of responders with constant communication delay and constant energy consumption. Poid can further identify the responders by elaborately-designed power assignment algorithm and also consume constant energy. We evaluate the performance of Poc and Poid in different wireless environments and explore the impact of different network characteristics. Results show that Poc and Poid provide fast accurate counting and identification with substantially lower delay than the state-of-the-art solutions.

Acknowledgments

The authors want to thank the reviewers for their constructive comments. This work was supported in part by China NSF grants (61472252, 61133006, 61321491, 61103224, 61371124, 61472445) and China 973 projects (2014CB340303, 2012CB316201). The preliminary result [31] was published at ACM/IEEE IPSN 2014.

References

[1] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," *Wireless Netw.*, vol. 8, pp. 481–494, 2002.

[2] P. Dutta, R. Musaloiu-e, I. Stoica, and A. Terzis, "Wireless ack collisions not considered harmful," presented at the 7th Workshop Hot Top. Netw., Alberta, BC, Canada, 2008.

[3] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, "Fidelity and yield in a volcano monitoring sensor network," in *Proc. 7th Symp. Oper. Syst. Design Implementation*, 2006, pp. 381–396.

[4] M. H. Ammar and G. N. Rouskas, "On the performance of protocols for collecting responses over a multiple-access channel," in *Ann. Joint Conf. IEEE Comput., Commun. Soc.*, 1991, pp. 1490–1499.

[5] W. Zeng, A. Arora, and K. Srinivasan, "Low power counting via collaborative wireless communications," in *Proc. 12th Int. Conf. Inf. Process. Sens. Netw.*, 2013, pp. 43–54.

[6] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh, "Efficient network flooding and time synchronization with glossy," in *Proc. Int. Conf. Inf. Process. Sens. Netw.*, 2011, pp. 73–84.

[7] M. Doddavenkatappa, M. C. Chan, and B. Leong, "Splash: Fast data dissemination with constructive interference in wireless sensor networks," in *Proc. 10th USENIX Conf. Netw. Syst. Design Implementation*, 2013, pp. 269–282.

[8] M. Ettus, *Universal Software Radio Peripheral (USRP)*, Ettus Research LLC, Santa Clara, CA, USA, 2008.

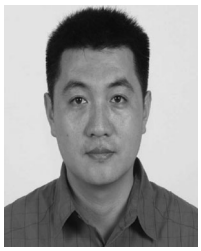
[9] *Node, TelosB*, Crossbow Inc., Milpitas, CA, USA, 2010.

[10] IEEE Std. 802.15.4-2003, 2003.

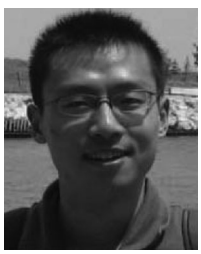
[11] W. Yin, Y. Liu, Y. He, X. Li, and D. Cheng, "Disco: Improving packet delivery via deliberate synchronized constructive interference," *IEEE Trans. Parallel Distrib. Syst*, vol. 26, no. 3, pp. 713–723, Mar. 2015.
[12] J. H. Davies, *MSP430 Microcontroller Basics*. New York, NY, USA: Elsevier, 2008.
[13] A. Farina and L. Ortenzi, "Effect of ADC and receiver saturation on adaptive spatial filtering of directional interference," *Signal Process.*, vol. 83, pp. 1065–1078, 2003.
[14] O. Kallenberg, *Foundations of Modern Probability*. New York, NY, USA: Springer, 2002.
[15] M. Hefeeda and M. Bagheri, "Forest fire modeling and early detection using wireless sensor networks," *Ad Hoc & Sens. Wireless Netw.*, vol. 7, pp. 169–224, 2009.
[16] A. Hasler, I. Talzi, J. Beutel, C. Tschudin, and S. Gruber, "Wireless sensor networks in permafrost research-concept, requirements, implementation and challenges," in *Proc. 9th Int. Conf. Permafrost*, 2008, pp. 669–674.
[17] C.-Y. Chong and S. Kumar, "Sensor networks: Evolution, opportunities, and challenges," *Proc. IEEE*, vol. 91, no. 8, pp. 1247–1256, Aug. 2003.
[18] R. C. Shah, S. Roy, S. Jain, and W. Brunette, "Data MULEs: Modeling and analysis of a three-tier architecture for sparse sensor networks," in *Proc. IEEE Int. Workshop Sens. Netw. Protocols, Appl.*, 2003, pp. 30–41.
[19] Y. Zeng, S. Ó. Murphy, L. Sitanayah, T. Tabirca, T. Truong, K. Brown, and C. Sreenan, "Building fire emergency detection and response using wireless sensor networks," presented at the 9th IT and T Conf., Dublin, Ireland, 2009.
[20] Y. Zhang and W. Lee, "Intrusion detection in wireless ad-hoc networks," in *Proc. 6th Annu. Int. Conf. Mobile Comput. Netw.*, 2000, pp. 275–283.
[21] (2013). UCLA ZigBee PHY. [Online]. Available: https://www.cgran.org/wiki/UCLAZigBee
[22] *CC2420 Datasheet*, Texas Instruments, Dallas, TX, USA, 2007.
[23] A. Dutta, D. Saha, D. Grunwald, and D. Sicker, "SMACK: A smart acknowledgment scheme for broadcast messages in wireless networks," in *Proc. ACM SIGCOMM Conf. Data Commun.*, 2009, pp. 15–26.
[24] D. Saha, A. Dutta, D. Grunwald, and D. Sicker, "PHY aided MAC—A new paradigm," in *Proc. IEEE INFOCOM*, 2009, pp. 2986–2990.
[25] M. Kodialam and T. Nandagopal, "Fast and reliable estimation schemes in RFID systems," in *Proc. 12th Annu. Int. Conf. Mobile Comput. Netw.*, 2006, pp. 322–333.
[26] M. Shahzad and A. X. Liu, "Every bit counts: Fast and scalable RFID estimation," in *Proc. 18th Annu. Int. Conf. Mobile Comput. Netw.*, 2012, pp. 365–376.
[27] H. Han, B. Sheng, C. C. Tan, Q. Li, W. Mao, and S. Lu, "Counting rfid tags efficiently and anonymously," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.
[28] H. Adam, E. Yanmaz, and C. Bettstetter, "Contention-based estimation of neighbor cardinality," *IEEE Trans. Mobile Comput.*, vol. 12, no. 3, pp. 542–555, Mar. 2013.
[29] Y. Wang, Y. He, X. Mao, Y. Liu, Z. Huang, and X. Li, "Exploiting constructive interference for scalable flooding in wireless networks," in *Proc. IEEE INFOCOM*, 2012, pp. 2104–2112.
[30] S. Yu, X. Wu, P. Wu, D. Wu, H. Dai, and G. Chen, "Cirf: Constructive interference-based reliable flooding in asynchronous duty-cycle wireless sensor networks," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2014, pp. 2734–2738.
[31] D. Wu, C. Dong, S. Tang, H. Dai, and G. Chen, "Fast and fine-grained counting and identification via constructive interference in WSNs," in *Proc. 13th Int. Symp. Inf. Process. Sens. Netw.*, 2014, pp. 191–202.

Dingming Wu received the BE degree from the Department of Computer Science, Wuhan University, China, in 2012. He is currently working toward the master's degree at the Department of Computer Science, Nanjing University, China. His current research interests include mobile computing and networked systems. His work incorporates both theoretical analysis and building practical systems. He worked as a Research Intern at Microsoft Research Asia in 2014. He is a student member of the IEEE.

Guihai Chen received the BS degree in computer software from Nanjing University, China, in 1984, the ME degree in computer applications from Southeast University in 1987, and the PhD degree in computer science from the University of Hong Kong in 1997. He is currently a professor and deputy chair of the Department of Computer Science, Nanjing University. He had been invited as a visiting professor by many foreign universities including the Kyushu Institute of Technology, Japan, in 1998, University of Queensland, Australia, in 2000, and Wayne State University, USA, from September 2001 to August 2003. He has a wide range of research interests with focus on sensor networks, peer-to-peer computing, high-performance computer architecture, and combinatorics. He is a member of the IEEE.



Chao Dong received the PhD degree in communication engineering from the Institute of Communication Engineering, Nanjing, China, in 2007. He is currently an associate professor at the College of Communication Engineering, PLA University of Science and Technology, China. From 2008 to 2011, he was a postdoc in the Department of Computer Science and Technology, Nanjing University, China. His current research interests include wireless cognitive networks, software-defined networks, and network coding. He is a member of the IEEE.



Shaojie Tang received the BS degree in radio engineering from Southeast University, China, in 2006, and the PhD degree from the Department of Computer Science, Illinois Institute of Technology, in 2012. He is currently an assistant professor in the Department of Information Systems, University of Texas at Dallas. His main research interests focus on wireless networks (including sensor networks and cognitive radio networks), social networks, security and privacy, and game theory. He has served on the editorial board of the *Journal of Distributed Sensor Networks*. He also served as TPC member of a number of conferences such as ACM MobiHoc, IEEE ICNP, and IEEE SECON. He is a member of the IEEE.

Haipeng Dai received the BS degree from the Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2010, and the PhD degree from the Department of Computer Science and Technology, Nanjing University, Nanjing, China, in 2014. He is currently a research assistant professor in the Department of Computer Science and Technology, Nanjing University. His research interests are mainly in the areas of wireless sensor networks and wireless networks. His research papers have been published in many prestigious conferences and journals such as IEEE INFOCOM, IEEE ICDCS, and IEEE TPDS. He serves/ed as Poster Chair of the IEEE ICNP'14, TPC member of the IEEE ICC'14 and the IEEE Globecom'14. He is a member of the IEEE and the ACM.