Fast and Fine-grained Counting and Identification via Constructive Interference in WSNs

Dingming Wu*, Chao Dong§, Shaojie Tang‡, Haipeng Dai*, and Guihai Chen*†
*State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
‡Department of Computer and Information Science, Temple University, Philadelphia, PA
†Shanghai Key Laboratory of Scalable Computing and Systems, Shanghai Jiao Tong University, China
§PLA University of Science and Technology, Nanjing, China
Emails:{dmwu0506,dch999,dhpphd2003}@gmail.com, shaojie.tang@temple.edu, gchen@nju.edu.cn

***Abstract*—Counting and identifying neighboring active nodes are two fundamental operations in wireless sensor networks (WSNs). In this paper, we propose two mechanisms, *Power based Counting* (Poc) and *Power based Identification* (Poid), which achieve fast and accurate counting and identification 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 of constructive interference (CI). However, due to the phase offsets and various hardware limitations (e.g., ADC saturation), the increased superposed power exhibits dynamic and diminishing returns as the number of component signals increases. This uncertainty of phase offsets and diminishing returns property of the superposed power pose serious challenges to the design of both Poc and Poid. To overcome these challenges, we design delay compensation methods to reduce the phase offset of each component signal, and propose a novel probabilistic estimation technique in cooperation with CI. We implement Poc and Poid on a testbed of 1 USRP and 50 TelosB nodes, the experimental results show that the accuracy of Poc is above 97.9%, and the accuracy of Poid is above 96.5% for most cases. In addition to their high accuracy, our methods demonstrate significant advantages over the state-of-the-art solutions in terms of substantially lower energy consumption and estimation delay.**

I. Introduction

Numerous applications in WSNs require the sensor nodes to be aware of their neighbors' up-to-date states to perform decision making. For example, nodes are usually interested in (1) the number of their neighbors in a particular state, e.g., how many of my neighbors have a battery level above a certain threshold [4], or even (2) the identities of their neighbors in a particular state, e.g., which nodes who have witnessed a common event [25]. In this sense, counting and identification are two fundamental operations in WSNs. Unfortunately, current WSN systems and protocols are not tailored to address the challenges of achieving fast and fine-grained counting and identification. Traditional counting and identification methods typically adopt a TDMA-based scheme. This shceme requires a poller to contact neighbors sequentially or reserve a unique time slot for each neighbor to respond [3]. This approach is inefficient since the poller's communication delay and energy consumption increase linearly with the network size. Recently, Zeng *et al.* [26] proposed two RSSI-based counting schemes, LinearPoll and LogPoll, which allow neighbors to respond simultaneously. However, LinearPoll has just constant energy consumption improvement over the TDMA-based scheme and LogPoll can only count nodes 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 [11] 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. Next, based on the power prediction 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, works in sparse networks, assigns diverse responding power for each neighbor and ensures that the received power from any two neighbors or any two combinations of neighbors are sufficiently different. Thus Poid can estimate the set of responders efficiently by identifying the unique received power. Conversely, Poc ensures that the received power from each neighbor is nearly the same, then it counts the nodes by identifying the power gain of a superposed signal relative to a single signal. Poc adopts a novel probabilistic estimation technique and is thus resilient to network density. Since the received power is processed locally, both these two mechanisms have constant communication delay, thus achieving fast counting and identification with constant energy cost.

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 [26], power superposition from multiple neighbors is rather difficult to predict since the POs across different transmissions are quite random. This uncertainty poses a serious challenge to the

design of both Poc and Poid. To handle this challenge, we take a close look at different types of delays in the processing of the neighbors' radios. We show that by compensating software delay, hardware delay and signal propagation delay across different nodes in an offline manner, PO can be reduced to a level of $0.25\,\mu$s with very small variability. The second challenge is that there exists an upper bound on the observed received power due to various hardware limitations, e.g., the poller's analog-to-digital converter (ADC) saturation. As a result, the observed received power of a superposed signal cannot truly reflect the real power if the real power exceeds a threshold. To address this challenge, we assign a response probability p to each node, then nodes will respond to the poller with probability p such that the received power is strictly 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.

The major contributions of this paper can be summarized as follows. (1) We investigate 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. (2) Based on this model, we propose Poc, a fast and fine-grained counting scheme that is resilient to network topology and size. Poc improves the counting accuracy from logarithmic scale of LogPoll to linear scale. (3) We further propose an efficient identification scheme, called Poid, based on this model. Poid achieves accurate identification with constant delay and reduces the energy consumption of LinearPoll from $O(n)$ to $O(1)$ in sparse networks. (4) We implement these two schemes on a testbed of 1 Gnuradio/USRP1 [9] and 50 TelosB [18] nodes. Experimental results show that the accuracy of Poc is above 97.9%, and the accuracy of Poid exceeds 96.5% for most cases. In addition, our methods achieve substantially lower energy consumption and estimation delay compared with the state-of-the-art solutions.

II. Superposed Signal with CI

A key insight behind our design is that the received power of a superposed signal increases with the number of responders given that 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.

A. Signal Power under CI

The IEEE 802.15.4 standard [15] compatible radios equipped in sensor nodes adopt an offset quadrature phase-shift keying (O-QPSK) modulation scheme. With half-sine pulse shaping, this scheme is equivalent to minimum-shift keying (MSK) [22]. 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 $0.5\,\mu$s [11].

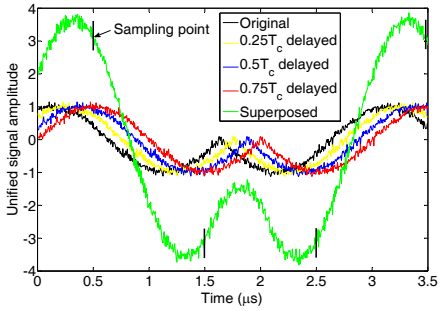


Fig. 1: Waveform of a superposed signal and four component signals.

Consider multiple signals superpose at the poller and each with noise $N_i(t)$. Let $s_r(t)$ denote the received superposed signal from n responders, A_i and τ_i are the amplitude and phase offset of the i^{th} signal. Then $s_r(t)$ 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 $2\,T_c$. According to [23], we get the peak amplitude A_r of the received signal (i.e., sample values of positive chips) as $A_r = \sum_{i=1}^{n} \left[A_i \cos(\frac{\pi}{2T_c}\tau_i) + n_i\right]$, where n_i represent the noise in the i^{th} signal after demodulation. Since signal power is proportional to the square of amplitude, by assuming that the proportionality constant is 1, we can get the received power P_r as

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

In Fig. 1, a 4-chips ([1 0 0 1]) MSK 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 is shown in Fig. 1, 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.

B. Beyond CI

CI requires that the MPO does not exceed $0.5\,\mu$s [11]. However, such precise synchronization may not be achieved by a simple common trigger. In [11], Ferrari *et al.* showed that by compensating software processing delays and mitigating hardware delays across different neighbors, CI can be achieved in commodity sensor platforms equipped with CC2420 radios. Nevertheless, this method alone can not ensure sufficient power gain of the superposed signals. Consider an extreme case where all delayed signals have phase offsets of $0.5\,\mu$s, P_r is equal to the power of a single signal.

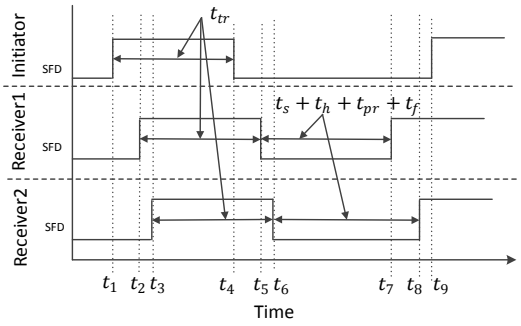


Fig. 2: Execution timeline of a initiator and two receivers with respect to the transitions of SFD pins.

1) Compensating Signal Propagation delay : We first take a closer look at the delays of different nodes that lead to the phase offsets among different transmissions.

Signal propagation delay t_p is the duration of signal's two-way around flight time between an initiator and a receiver. Strictly speaking, there also exists delay in the radio's sensing an arriving transmission (i.e., the time required by radio 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 radios'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 radio detects the transmission request from the MCU.

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

In Fig. 2, we plot the SFD activities of 1 initiator and 2 receivers and specify these 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. These two delays only depend on the standard. In [11], Ferrari *et al*. showed that 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. Hereafter, we assume $t_s + t_h + t_{pr} + t_f$ 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 [23].

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. This fact is also used in [23]. Define by λ_i the clock drift coefficient of node i relative to the initiator. Then in Fig. 2 we have $\lambda_1 = \frac{t_4 - t_1}{t_5 - t_2}, \quad \lambda_2 = \frac{t_4 - t_1}{t_6 - t_3}$. To estimate λ_i, the initiator 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 signal propagation delay for each node: $t'_{pi} = t_c^0 - t_c^i \lambda_i$. This process can be repeated for multiple rounds to get the average t'_{pi} to reduce variance. In real applications, the clock drift coefficients need to 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 [6]. 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) Experiments on Synchronization: We randomly selected 31 TelosB nodes in our testbed shown in §. VII. One of them acts as the initiator and the remainder are responders. The initiator's SFD pin is connected to a logic analyzer with a granularity of 2 ns. In the initial stage, we let the initiator perform 30 packet exchanges with each neighbor to compensate their respective propagation delays. Then the experiment proceeds in rounds. In each round, the initiator 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. We initialize k with 2 and sequentially increase it by 2 until $k = 30$. In each round, we repeat the experiment 200 times.

Fig. 3 compares the synchronization accuracy of Glossy in [11] and the one after adding the propagation delay compensation (PDC). Dots on the lines shows the average phase offsets; error bars indicate the standard deviation. The results 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 μs 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 μs.

Fig. 4 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

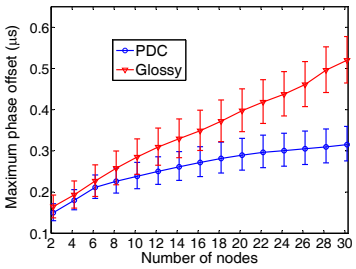


Fig. 3: Synchronization error against different number of simultaneous transmissions.

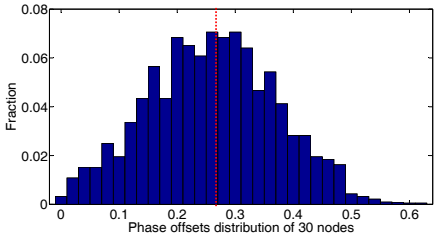


Fig. 4: Distribution of phase offsets among 30 nodes. Results are obtained by computing the difference of each t_c^0 relative to the smallest one. Mean value (dotted line) is 0.268 μs.

(as re-selecting the 30 responders in the testbed shown in § VII). 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 90^{th} 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 power based counting and identification, which we will demonstrate in the following.

C. Limitations on Power Superposition

From the experiment results in § II-B2, it is possible to conclude that if we add a transmission with unit amplitude, the amplitude gain of the superposed signal would be $\cos(\frac{\pi}{2} \cdot \frac{0.4\,\mu s}{T_c}) = 0.31$ with high probability because the extreme phase offsets (either too large or too small) are rare. 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 shown in Fig. 3, MPO across different transmissions increases with the number of concurrent transmitters[1]. We cannot ensure that MPO is always below a certain value with more responders. (2) Constrained by the saturation rate of ADC built in the initiator, sample values exceeding a threshold will be truncated [10], which indicates that there exists an upper bound on the observed received power of a superposed signal.

We conduct an experiment to evaluate the observed received power of superposed signals under the same setting as in § II-B2. The difference is that in each round, the initiator will trigger all the selected neighbors to perform a transmission simultaneously at the maximum power level. The initiator records the received power in each round. As shown in Fig. 5, the increasing tendency of received power starts to slow down when more than 24 responders respond. 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 initiator's ADC. This threshold is usually lower than the initiator's nominal saturation value due to the non-linearity effect of ADCs [10]. Based on the results in Fig. 5, in our following experiments, we set $P_0 = 1.2$ mW.

[1]This problem is also validated in [7].

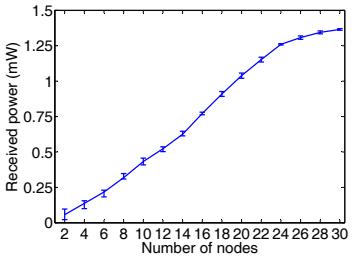


Fig. 5: Received power against different number of simultaneous transmissions.

III. System Model

A. Single Signal Power Prediction

We consider a static network composed of a poller s and a neighbor set N. Denote by $L(s,i)$ the path loss from 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 unless external interference occurs. Let $P(i,h_i)$ be the received power obtained by s when a single neighbor i responds at power h_i. We have $P(i,h_i) = a_i h_i + b_i$, where a_i and b_i are coefficients that we should estimate, b_i depends on both the additive background noise $\mathcal{N}$ and the path loss $L(s,i)$ [26]: $b_i = \mathcal{N} - L(s,i)$.

To predict $P(i,h_i)$ under a given 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 $\widehat{b}_i$, we compute the estimation of path loss as $\widehat{L}(s,i) = \mathcal{N} - \widehat{b}_i$. 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.

Note that we need only to perform the above power prediction once, and this can be done during the node synchronization phase (see § II-B) at no additional cost.

B. Power Superposition Model

Here we introduce the prediction on the received power of superposed signals. Let I be a non-empty subset of N, $P(I)$ be the received power of the superposed signal from nodes in I. By substituting the amplitude A_i in Eq. 1 with $\sqrt{\widehat{P}(i,h_i)}$, we obtain the estimation of $P(I)$ as

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

where signal k is the earliest signal reaching s and 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 $\widehat{P}(i,h_i)$.

Now, phase offset τ_i is the only unknown parameter. Nevertheless, it appears rather difficult to estimate such a random parameter. By compensating propagation delay for each node in § II-B1, we show that τ_i can be confined to a smaller range from $0.1\,\mu$s to $0.4\,\mu$s with high probability. For simplicity, we use the average case of $0.25\,\mu$s to all delayed signals. Thus, we can rewrite Eq. 2 as

$$\widehat{P}(I) = \left[\sqrt{\widehat{P}(k,h_k)} + \cos\frac{\pi}{4}\sum_{i\in I\setminus k}\sqrt{\widehat{P}(i,h_i)}\right]^2. \quad (3)$$

We have the following requirement for bounded prediction error by using the above estimation: *There exists some δ, for any non-empty set $I \subseteq N$: $\widehat{P}(I) - \delta \le P(I) \le \widehat{P}(I) + \delta$.* Based on our experiments with CC2420 radios, we observe that the 99^{th} percentile of δ is at most 0.03 mW. Therefore, this requirement is satisfied in a static network if no interference occurs.

IV. Poc design

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

A. 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. We may repeat this process for multiple rounds until achieving a satisfactory accuracy. 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 $x_i\,(i = 1, ...m)$ 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 $x_i \le n_0$ 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 $n \le n_0$, by letting $p = 1$, the poller can directly get $\hat{n}$ without additional operations. Avoiding trivial cases, we only consider the case where $n \in (n_0, |N|]$ in the following.

B. Response Power Assignment

Prior to entering the counting process, poller s first assigns each node a transmitting power such that the received power of each responder is similar at the poller. We assume that the neighbors are labeled in the order of increasing path loss. Let α be the received power of the first neighbor transmitting with the minimum power level, β be the received power of the last neighbor transmitting with the maximum power level. Denote the common received power as ξ, clearly, if $\alpha > \beta$, no such ξ exists. If $\alpha \le \beta$, we compute ξ by minimizing the sum of power adjustment gap over all neighbors: $\xi = \text{argmin}_{t\in[\alpha,\beta]}\left(\sum_{i\in N}|P(i,h_i) - t|\right)$. This is similar to the methods used in [26].

To ensure that different number of responders generate distinguishable 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. 3, we construct the relation between $P(I)$ and the cardinality of I:

$$P(I) = \left(\sqrt{\xi} + \frac{\sqrt{2}}{2}(|I| - 1)\sqrt{\xi}\right)^2. \quad (4)$$

Since the function of the righthand side of Eq. 4 is monotonous, it has unique inverse, denoted by $g(\cdot)$. Then we can get the cardinality of I by $g(P(I))$. Apparently, n_0 can be computed as $g(P_0)$.

C. Estimation Methodology

1) Estimator: Compared with traditional independent binomial random samples ($B(n,p)$), the probability of getting a sample whose value is no larger than n_0 (i.e. $0, 1, 2, \cdots, n_0$) remains the same, but the probability of obtaining a sample greater than n_0 is 0 due to the upper bound on the received power. Therefore, the probability mass function needs to be normalized to reflect a smaller universe. In other words:

$$p_k = \begin{cases} \frac{\binom{n}{k}p^k(1-p)^{n-k}}{p_c} & 0 \le k \le n_0 \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

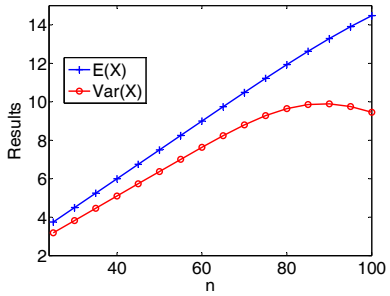
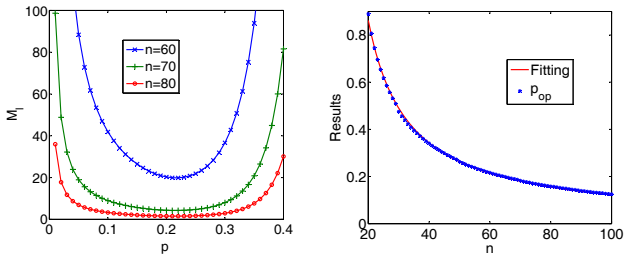


Fig. 6: Expected value and variance of X against different responder size n under $p = 0.15,\ n_0 = 20$.



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

Fig. 7: Convexity of M_l with respect to p and the corresponding p_{op} for different number of n. Results are obtained under $\theta = 0.05$, $\eta = 0.95$.

where p_k is the probability that $x_i = k$, p_c is the probability that $x_i \leq n_0$. 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: $E(X) = \sum_{k=0}^{n_0} k p_k, \mathrm{Var}(X) = E(X^2) - E(X)^2 = \sum_{k=0}^{n_0} k^2 p_k - (\sum_{k=0}^{n_0} k p_k)^2$. In Fig. 6, we plot the expected value and variance of X with respect to different n with $p = 0.15,\ n_0 = 20$. As is shown in Fig. 6, $\mathrm{Var}(X)$ is not a monotonous function. Hence, it can not be used as the estimator. On the other hand, we see $E(X)$ is a monotonous increasing function of n. This is intuitive since under 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 $\bar{X}$ is averaged over m independent measurements, its variance becomes $\mathrm{Var}(X)/m$.

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 $P\{|\hat{n} - n| \leq \theta n\} \geq \eta$. 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

$$\begin{aligned} &P\{(1-\theta)n \leq \mu^{-1}(\bar{X}) \leq (1+\theta)n\} \\ &= P\{\mu((1-\theta)n) \leq \bar{X} \leq \mu((1+\theta)n)\} \geq \eta. \end{aligned} \tag{6}$$

Let $Y = \frac{\bar{X}-\mu(n)}{\sqrt{\sigma(n)}}$, then (6) can be rewritten as

$$P\left\{\frac{\mu((1-\theta)n)-\mu(n)}{\sqrt{\sigma(n)}} \leq Y \leq \frac{\mu((1+\theta)n)-\mu(n)}{\sqrt{\sigma(n)}}\right\} \geq \eta. \tag{7}$$

We know that Y approximates a standard normal random variable based on the central limit theorem. 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 $\theta \leq 0.1,\ n \leq 200$. Thus, it is safe to assume that $y_l = -y_u$, thereafter, indicating that the confidence interval requirement is symmetric on both the upper and lower sides of n. Applying this symmetry property and $\sigma(n) = \mathrm{Var}(X)/m$ into (7) and rearranging for m yields

$$m \geq \frac{\mathrm{Var}(X)[\Phi^{-1}(\frac{1+\eta}{2})]^2}{(\mu((1+\theta)n) - \mu(n))^2} \tag{8}$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution. Inequality 8 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} denote the right hand side of Inequality 8, then we compute the lower bound of the expected value of M as

$$M_l = \frac{m_{min}}{p_c} = \frac{\mathrm{Var}(X)[\Phi^{-1}(\frac{1+\eta}{2})]^2}{(\mu((1+\theta)n) - \mu(n))^2 p_c}. \tag{9}$$

Note that M_l is the estimation delay, our goal is to find an optimal response probability p_{op} in order to minimize M_l. In Fig. 7(a), we show that M_l is a convex function with respect to p under different values of n, thus such an optimal value p_{op} always exists. In Fig. 7(b), 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 $p_{op}(n) = 11.35/(n-7)$.

3) Two-phase Estimation: As shown in Fig. 7(b), 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 a two-phase estimation method to adaptively estimate n: the poller first performs a coarse estimation $\hat{n}$ using a randomly selected p over range $(n_0, |N|]$; in the second phase, $\hat{n}$ will be used as feedback to adjust the response probability adopted in the previous phase. By following this strategy, we can find p_{op} eventually.

Phase one. Our former analysis only considers effective samples. An ineffective sample, however, also contains useful information about the parameter n. In this phase, we take both the effective and ineffective samples into consideration and use the maximum likelihood estimation (MLE) to give a relatively "accurate" coarse estimation of n.

Assume that we obtain M_1 samples among which a samples are ineffective and b samples are effective. Effective samples are denoted as $x_1, x_2, ...x_b$. We know that the probability of an ineffective sample being observed is $1 - P\{0 \leq X \leq n_0\} = 1 - p_c$. Thus, the likelihood function $L(n)$ is expressed as

$$L(n) = (1-p_c)^a \cdot \prod_{i=1}^{b} \left[\binom{n}{x_i} p^{x_i} (1-p)^{n-x_i} \right]. \quad (10)$$

Estimation $\hat{n}$ is returned as the argument over $(n_0, |N|]$ that maximize $L(n)$.

Phase two. Poller s then uses $p_{op}(\hat{n})$ to perform an accurate estimation. s will first compute the number of effective samples required to satisfy a desired accuracy $1-\theta$ with confidence η (see Eq. 8). Then s will launch the measurements to get m_{min} effective samples. At last, s estimates n based on the sample mean $\bar{X}$.

Algorithm 1 summarizes the whole estimation procedure.

Algorithm 1: Two-phase Estimation

Input: $n_0, N, \theta, \eta, M_1$
Output: $\hat{n}$

Get received power P_r using $p = 1$
if $P_r \leq P_0$ **then**
  return $g(P_r)$
else
  $\hat{n} \leftarrow$ **CoarseEstimate**(n_0, N, M_1)
  $\hat{n} \leftarrow$ accurate estimation with $p = p_{op}(\hat{n})$
  return $\hat{n}$
CoarseEstimate(n_0, N, M_1)
$n' \leftarrow rand(n_0, N)$
Get M_1 samples using $p_{op}(n')$, stored in S
$\hat{n} \leftarrow MLE(n_0, N, p_{op}(n'), S)$
return $\hat{n}$

D. Estimation Delay Evaluation

We evaluate the estimation delay of two-phase estimation scheme (TES) in Matlab. For ease of comparison, we also propose a baseline in which the poller s uses $p = p_{op}(|N|)$ for estimation and keeps p unchanged in all measurements. In the evaluation, we set $n_0 = 20$, $|N| = 100$, and a varying M_1 to evaluate the impact of coarse estimation.

Note that the expected number of measurements of TES contains two parts, caused by two phases respectively. In the first phase, the number of measurements is fixed as M_1. In the second phase, the expected number of measurements of TES is computed in the same way of the baseline: both obtained by plugging their respective p_{op} into Eq. 9.

We plot the results in Fig. 8, each result of TES is averaged over 2000 rounds. It is observed that TES's delay increases with M_1, indicating that the coarse estimation based on one single sample is sufficient to find an optimal p. Therefore, we use $M_1 = 1$ in our following experiments. For $\theta = 0.1$, $\eta = 0.9$, we observe that TES is on average 1.5 times faster than the baseline. For $\theta = 0.05$, $\eta = 0.95$ and $\theta = 0.01$, $\eta = 0.95$, TES is 2 times faster and 8 times faster than the baseline respectively.

V. 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.

A. 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; (3) Identify the nodes that have responded. Notice that the first step is done offline for one-time only while the second and the third steps are two routines that can be finished with constant delay.

B. Response Power Assignment

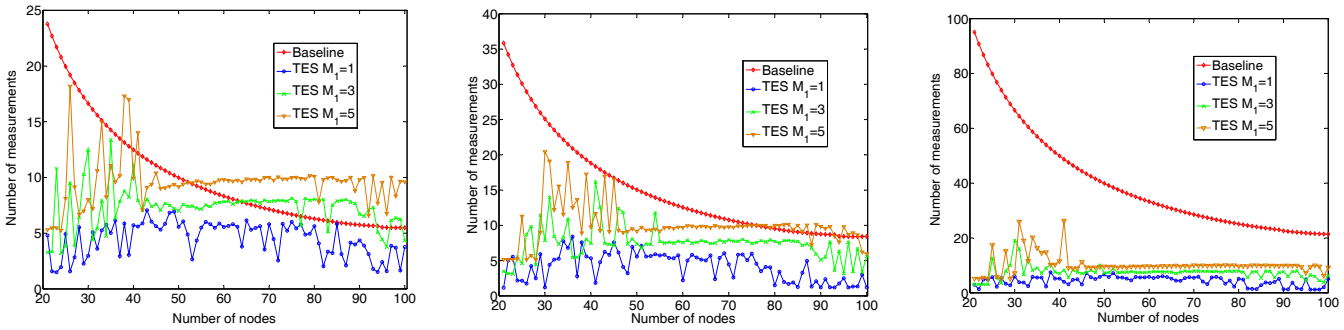
Unlike in Poc, in Poid, nodes are assigned diverse power levels such that any individual or combination of responders will generate unique received power at the poller. We propose a simple yet efficient algorithm in Alg. 2 to achieve this goal.

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 $I \leftarrow \emptyset$
4 $h_1 \leftarrow \text{argmin}_{h \in H}(\widehat{P}(1,h) - P_l \geq \delta)$
5 $I \leftarrow I \bigcup$ node 1
6 **for** $i \leftarrow 2$ *to* $|N|$ **do**
7 **for** $k \leftarrow 1$ *to* $|H|$ **do**
8 **if** $\widehat{P}(i, H_k) - \widehat{P}(I) > 2\,\delta$ **then**
9 $h_i = H_k$
10 $I \leftarrow I \bigcup$ node i
11 **break**

In Alg. 2, we compute an assignment $\{h_i\}$ for each node $i \in N$ using a simple rule: the larger the path loss of node i, the lower the assigned power. In this way, we ensure that the received power is sufficiently separated. Line 4 assigns the minimum transmitting power to node 1 such that $\widehat{P}(1, h_1) - \delta$



(a) $\theta = 0.1,\ \eta = 0.9$ (b) $\theta = 0.05,\ \eta = 0.95$ (c) $\theta = 0.01,\ \eta = 0.95$

Fig. 8: Number of measurements (delay) against different responder size.

is above P_l, the weakest response power that can be distinguished by the poller. For each of the remaining nodes, it searches for a minimum power level such that the received power of the current node is at least $2\,\delta$ above $\widehat{P}(I)$ where I contains all the nodes whose transmitting power has been assigned in the former iterations. The condition in line 8 is a key factor for maintaining the power gap among different combinations of nodes.

C. Finger-Print based Identification

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

D. Discussions

Due to the hardware limitations presented in § II-C, Poid only applies to sparse networks with relatively small neighborhood size. In sparse networks, we can carefully calculate the power levels for each node to ensure that the aggregate power of all transmissions will never exceed the power threshold of the poller. Sparse networks are often encountered since in many cases limited sensors need to be spread over a large geographical area and dense deployments may not be cost effective and practical. Examples include forest fire monitoring [14], permafrost monitoring [13] and battlefield monitoring [5]. In these scenarios, a sink or a mobile agent in the case of Data MULEs [20] usually has small neighborhood size, thus Poid can be applied on them to achieve efficient node identification.

VI. Implementation

We implement Poc and Poid on a testbed consisting of 1 GNURadio USRP with RFX2400 daughterboard and 50 TelosB nodes. The USRP runs the UCLA ZigBee PHY [1] 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 [16].

A. Received Power Measurement

The 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. 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 poller 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 sensed a energy elevation that is above 0.04 mW, it starts recording the power samples.

B. Handling External Interference

We consider three cases where interference can play and introduce corresponding handling mechanisms in the power measurement procedure.

False positive energy elevation. In this case, the energy elevation that s sensed is caused by interference rather than the transmissions of neighbors. 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) [16]. Let ϵ be the error tolerance[2] of t_Δ. 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 pollution. In this case, several peaks of S are polluted by random impulses or sporadic WiFi transmissions. 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 and

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

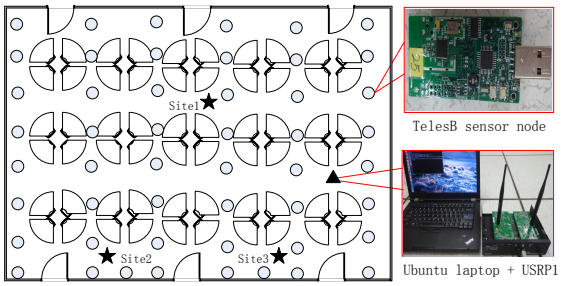
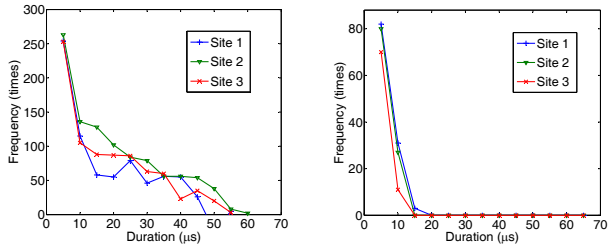


Fig. 9: Physical layout of 50 nodes and a USRP in a 18×25 m laboratory.



(a) Channel 25 (b) Channel 26

Fig. 10: Duration of interference in different locations with respect to different channels.

variance, denoted as μ and σ, respectively, 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 WiFi transmission. In this case, the transmissions of the neighbors are partially or entirely covered by a WiFi transmission which has longer duration than our transmissions. To detect this interference, 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.

C. Discussions of Using USRP

We choose to use a USRP to act as the poller because it can report power information with higher resolution than commodity sensors do. The scenario in this paper may not be a typical scenario of current WSNs. However, to perform efficient counting and identification, we do not need to modify any deployed nodes. All we need to do is add an additional node (a USRP in our cases) equipped with more powerful radios. Based on our experimental results, we argue that it is rewarding to add such a special node.

VII. Performance Evaluation

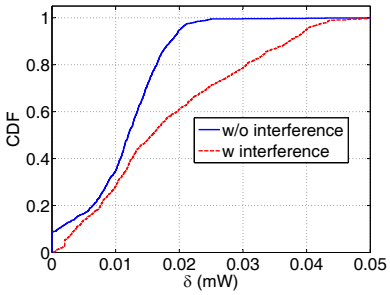
In this section, we evaluate the performance of Poc and Poid under different wireless environments and parameter settings. The testbed is located in a 18×25 m laboratory of a school building. As illustrated in Fig. 9, 50 sensor nodes are placed in a grid-like topology. The USRP is connected to a Ubuntu 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 the response packets and the observed received power, 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.

A. Micro Benchmark

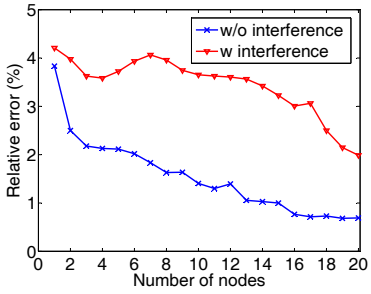
We conduct micro-benchmark to (1) investigate the interference occurs in the lab; (2) measure the distribution of power prediction error δ, and (3) evaluate Poc's performance when the responder size is no larger than n_0.

1) Investigating the Interference: Both Poc and Poid work on a channel that has little or no interference. On channels that experience enormous energy pulses like WiFi transmissions, Poc and Poid may suffer severe performance degradation. In the following, we choose channel 25 and 26 to evaluate the impact of "mild" interference since these two channels experience little WiFi energy. Channel 25 is chosen only for comparison, in real applications, channel 26 is recommended. To investigate the interference, we place a USRP in three sites (illustrated by three stars in Fig. 9). The USRP is programmed to listen on channel 25 and channel 26, respectively, for 30 seconds. Based on the power samples returned, we measure the duration distributions of interference on each channel. Results are illustrated in Fig. 10(a) and Fig. 10(b). We observe that the results from different sites are similar. In particular, results from both channel 25 and 26 show that short impulse (less than 10 μs) plays the main role of interference types. Note that 10 μs is less than the duration of an O-QPSK symbol (16 μs), implying that the interference type is mainly *partial peaks pollution*, which can be handled by the poller without remeasurement. Since channel 25 is close to channel 11 of WiFi communications, we observed more frequent and relatively longer interference. Further, for the percentage of interference in a period of 30 s, channel 25 sees 17% while channel 26 sees only 0.6%. In the following experiments, we regard channel 26 as an interference-free environment.

2) 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 200 times. The cumulative distribution function (CDF) of δ is plotted in Fig. 11(a). We see that in an interference-free environment (in channel 26), δ has 99^{th} percentile being at most 0.03 mW. Thus, the requirement



(a) Power prediction error



(b) Counting accuracy

Fig. 11: Evaluation of the power superposition model

of bounded prediction error is satisfied in an interference-free environment. Actually, in the strict sense, channel 26 is not a completely no-interference environment. Therefore, the percentile of 0.03 mW is supposed to be even larger than 99 in a perfectly no-interference environment. We also observe that this percentile is still above 80 in a environment of sporadic WiFi interference (channel 25), which may attribute to the mechanisms of handling interference adopted by the poller.

3) Counting with Controlled Responder Size: We then use $\delta = 0.03\,\text{mW}$ to evaluate the performance of Poc under limited responder size on different channels. Experiments are conducted under the same setting as in § VII-A2. Based on Eq. 4, we get $n_0 = 20$ here. Experiment results are plotted in Fig. 11(b), averaged over 200 estimations for each responder size. When there is no interference, the relative error is as low as 1.5% and is always below 4%. When interference occurs, the relative error shows just a 2% increase. Note that both of these two errors show a decreasing tendency when the number of nodes increases. We guess this is due to the decrease of power variation when responder size increases, which is also observed in Fig. 5.

B. Performance Evaluation of Poc and Poid

In this section, we evaluate the performance of Poc and Poid on the testbed and explore the impact of different factors.

1) Accuracy and Delay: For Poc, we evaluate it's counting accuracy under different number of samples. 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 [3]. 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 maybe 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 test different values of the total number of samples M (5, 10, 15) in which the first sample is used to get a coarse estimation and the rest $M-1$ samples are used to get an accurate estimation. We conduct 200 rounds of experiments for each number of responders varying from 21 to 50. The average error is summarized in Table I. We see that increasing M significantly improves the accuracy of Poc. For $M = 15$, the relative error is comparable with the baseline and below 2.5% on both channels. TDMA still cannot count 100 percent accurately because it is also impacted by the external interference. However, TDMA uses linear number of slots to the network size while Poc uses 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. Interference does impact the performance of Poc, but the performance degradation is always below 10%. Therefore, Poc is relatively robust to interference.

Poid. Nine nodes are used in this experiment. The responder set changes from round to round and all nodes in the testbed are covered. We compare Poid with LinearPoll [26], which is also a power based identification method 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 Λ of LinearPoll to be 4, which reports the best performance.

Table II summarizes the results. We use Ω to denote the set of responders, and then define counting error $e_{|\Omega|}$ and identification error e_Ω as: $e_{|\Omega|} = \frac{|\widehat{\Omega}|-|\Omega|}{|\Omega|}, e_\Omega = \frac{|\widehat{\Omega}\setminus\Omega|+|\Omega\setminus\widehat{\Omega}|}{|\Omega|}$.

We first examine the impact of the interference. For Poid, counting error $e_{|\Omega|}$ sees an increase of 8.4% when interference occurs, identification error e_Ω sees an increase of 7%. For LinearPoll, counting error sees an increase of 4.5% and identification error sees an increase of 3.1%. We then compare the performance of them on Ch.26, we see that counting error of Poid is comparable with LinearPoll and the identification error e_Ω is 0.4% larger than that of LinearPoll. These results show that LinearPoll is relatively more accurate and robust than Poid. This is due to the fact that LinearPoll uses both response length and received power to identify nodes, while nodes in Poid have identical minimum response length. However, by avoiding assigning different response length to different nodes, Poid shows significant less communication delay compared with LinearPoll.

		Relative error e (%)	
		Ch. 25	Ch. 26
Poc	M=5	10.2	9.7
	M=10	4.5	3.9
	M=15	2.4	2.1
TDMA		2.1	1.8

TABLE I: Average relative error of Poc and TDMA on different channels.

	$e_{\|\Omega\|}$ (%)		e_{Ω} (%)	
	Ch. 25	Ch. 26	Ch. 25	Ch. 26
Poid	2.3	2.1	3.8	3.5
LinearPoll	2.2	2.1	3.2	3.1

TABLE II: Average relative error of Poid and LinearPoll on different channels.

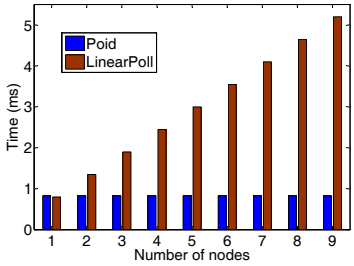
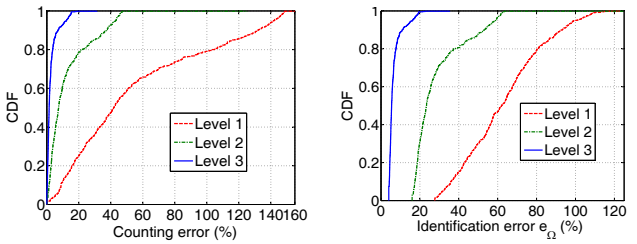
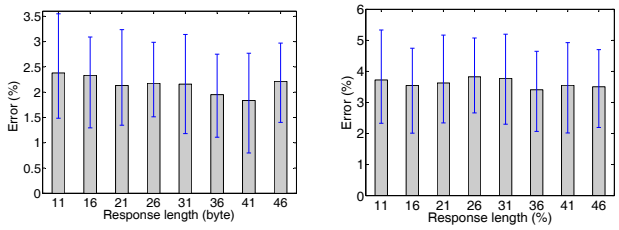


Fig. 12: Poller's radio on-time against different number of responders



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

Fig. 13: Impact of synchronization on the performance of Poc and Poid.



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

Fig. 14: Impact of packet length on the performance of Poc and Poid.

We then measure the poller's radio on-time for Poid and LinearPoll respectively in Fig. 12, each result is averaged over 200 rounds. Since in LinearPoll, the response lengths must be 4 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.

2) 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 [11].

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 1000 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. 13(a) plots the CDF of relative error of Poc under different synchronization levels. We observe that both of level 2 and level 3 show significant accuracy advantage over level 1. For 80^{th} percentile, the relative error is 5%, 20% and 100% for level 3, 2 and 1 respectively. For counting error's CDF of up to 20%, level 1 sees only 25% while level 2 sees 80% and level 3 sees 100%. We also observe similar results for Poid in Fig. 13(b). Therefore, Fig. 13 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% decrease on the average error of Poc and 21% decrease on the average error of Poid when compared with level 2.

3) 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 lengths. We re-conduct the experiments with eight different ACK lengths on channel 25 and the other settings are the same as in § VII-B2. Average error of Poc and Poid, plotted in Fig. 14, are both fairly constant and very similar (within 0.5% 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.

VIII. 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 [8], [19]. Although this scheme can achieve counting and identification in constant delay, it is not suitable for sensors for it's heavy computation burden brought by Fourier Transformations. Time slots based methods have been extensively studied in the field of counting RFID tags [2], [12], [17], [21]. 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 [26], 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 [11], [24], enhancement of overall link PRR [23] 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.

IX. CONCLUSIONS

This paper investigates the problem of counting and identification 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 energy consumption than the state-of-the-art solutions.

ACKNOWLEDGMENTS

We thank Xiaojun Zhu, Lizhao You, the reviewers and our shepherd, Luca Mottola for their insightful comments. This research is partly supported by China NSF grants (61133006, 61103224, 61371124, 61321491), China 973 projects (2014CB340300, 2012CB316200) and the NSF of Jiangsu Province under Grant No.BK2011118.

REFERENCES

[1] UCLA ZigBee PHY. https://www.cgran.org/wiki/UCLAZigBee.
[2] H. Adam et al. Contention-based estimation of neighbor cardinality. *IEEE Transactions on Mobile Computing*, pages 542–555, 2013.
[3] M. H. Ammar and G. N. Rouskas. On the performance of protocols for collecting responses over a multiple-access channel. In *INFOCOM*, 1991.
[4] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. *WINET*, 2002.
[5] C.-Y. Chong and S. P. Kumar. Sensor networks: evolution, opportunities, and challenges. *Proceedings of the IEEE*, 91(8):1247–1256, 2003.
[6] J. H. Davies. *MSP430 Microcontroller Basics*. Access Online via Elsevier, 2008.
[7] M. Doddavenkatappa, M. C. Chan, and B. Leong. Splash: fast data dissemination with constructive interference in wireless sensor networks. In *NSDI*, 2013.
[8] A. Dutta, D. Saha, D. Grunwald, and D. Sicker. Smack: a smart acknowledgment scheme for broadcast messages in wireless networks. In *ACM SIGCOMM*, 2009.
[9] M. Ettus. Universal software radio peripheral (usrp). *Ettus Research LLC http://www. ettus. com*, 2008.
[10] A. Farina and L. Ortenzi. Effect of adc and receiver saturation on adaptive spatial filtering of directional interference. *Signal processing*, 2003.
[11] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh. Efficient network flooding and time synchronization with glossy. In *IPSN*, 2011.
[12] H. Han, B. Sheng, C. C. Tan, Q. Li, W. Mao, and S. Lu. Counting rfid tags efficiently and anonymously. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9. IEEE, 2010.
[13] 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 Intl Conf. on Permafrost (NICOP 2008)*, volume 1, pages 669–674, 2008.
[14] M. Hefeeda and M. Bagheri. Forest fire modeling and early detection using wireless sensor networks. *Ad Hoc & Sensor Wireless Networks*, 7(3-4):169–224, 2009.
[15] IEEE. std. 802.15.4-2003. 2003.
[16] T. Instruments. Cc2420 datasheet, 2007.
[17] M. Kodialam and T. Nandagopal. Fast and reliable estimation schemes in rfid systems. In *MobiCom*, 2006.
[18] T. Node. Crossbow inc, 2010.
[19] D. Saha, A. Dutta, D. Grunwald, and D. Sicker. Phy aided mac-a new paradigm. In *INFOCOM 2009, IEEE*, pages 2986–2990. IEEE, 2009.
[20] R. C. Shah, S. Roy, S. Jain, and W. Brunette. Data mules: Modeling and analysis of a three-tier architecture for sparse sensor networks. *Ad Hoc Networks*, 1(2):215–233, 2003.
[21] M. Shahzad and A. X. Liu. Every bit counts: fast and scalable rfid estimation. In *MobiCom*, 2012.
[22] B. Sklar. *Digital communications*, volume 2. Prentice Hall NJ, 2001.
[23] Y. Wang, Y. He, D. Cheng, Y. Liu, and X.-y. Li. Triggercast: Enabling wireless collisions constructive. *arXiv preprint arXiv:1208.0664*, 2012.
[24] Y. Wang, Y. He, X. Mao, Y. Liu, Z. Huang, and X. Li. Exploiting constructive interference for scalable flooding in wireless networks. In *INFOCOM*, 2012.
[25] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh. Fidelity and yield in a volcano monitoring sensor network. In *OSDI*, 2006.
[26] W. Zeng, A. Arora, and K. Srinivasan. Low power counting via collaborative wireless communications. In *IPSN*, 2013.