Computer Networks 54 (2010) 658–673

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Physically-based models of low-power wireless links using signal power simulation ☆

Tal Rusak [a,b,*], Philip Levis [b]

[a] Department of Computer Science, Cornell University, Ithaca, NY 14853, USA
[b] Computer Systems Laboratory, Stanford University, Stanford, CA 94305, USA

ARTICLE INFO

Article history:
Available online 27 August 2009

Keywords:
Wireless link simulation
Low-power wireless networks
Sensor networks
Burstiness
Signal strength

ABSTRACT

We propose deriving wireless simulation models from experimental traces of radio signal strength. Because experimental traces have holes due to packet losses, we explore two algorithms for filling the gaps in lossy experimental traces. Using completed traces, we apply the CLOSEST-FIT PATTERN MATCHING (CPM) algorithm, originally designed for modeling external interference, to model signal strength.

We compare the observed link behavior using our models with that of the experimental packet trace. Our approach results in more accurate packet reception ratios (PRR) than current simulation methods, reducing the absolute error in PRR by up to about 0.3 in the experiments we present. We also find that using CPM for signal strength improves simulation of packet burstiness, reducing the Kantorovich–Wasserstein (KW) distance of conditional packet delivery functions (CPDFs) by a factor of about three for intermediate links. Our model reduces the factor of error in the number of parent changes in the standard TinyOS collection protocol (CTP) by an order of magnitude or more as compared to a real signal power trace in two simple test scenarios. We show that our methods are robust to the sampling frequency of the learning deployment and are thus generally applicable for simulating arbitrary applications without a pre-determined packet transmission frequency.

These improvements give low-power wireless network simulators a better capability to capture real-world dynamics and edge conditions that protocol designers typically must wait until deployment to detect.

1. Introduction

Many low-power wireless network deployments have observed significant differences between behavior in controlled environments such as test labs or simulation and behavior in the field. These differences can cause applications to fail at collecting the desired data [1]. Determining the cause of these failures is difficult due to the highly constrained nature of network hardware, including only a few bits of output information and little memory to save performance logs. The remote nature of many sensor network deployments make the study of the operation of the network and debugging protocols and applications even more difficult.

Increasing simulation fidelity, so simulators can capture edge cases and complexities encountered in real deployments, will reduce the gulf between testing and practice. It will also allow network developers to use the resources of a PC to debug and develop full scale applications. Creating such models will facilitate analytical studies regarding the nature of such systems. Low-power wireless networks

☆ A preliminary version of this manuscript appeared in ACM MSWiM'08.

* Corresponding author. Present address: Room 286 Gates Building, 353 Serra Mall, Stanford University, Stanford, CA 94305 USA.
E-mail addresses: rusakt@stanford.edu (T. Rusak), pal@cs.stanford.edu (P. Levis).

1389-1286/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2009.08.014

have proven difficult to simulate because of a large number of factors that impact their operation and a limited theoretical understanding. In particular, the precise modeling of the variation of noise and signal power when receiving a packet in low-power wireless networks is an open problem. A special challenge in creating these models is that many low-power wireless networks share the radio frequency (RF) environment, and especially the 2.4 GHz frequency range, with 802.11 wireless networks, microwaves, cordless telephones, and many other interference sources.

Wireless simulators have traditionally relied on analytical models for signal strength and noise. These models allow developers to explore a huge space of possible configurations and network designs and also enable a good understanding of packet dynamics. Such simulations have had success in modeling highly-complex wireless environments, for example, predicting the cell phone network's coverage of cities. For example, the WiSE tool [2] developed in the mid-1990's and several commercial tools available today [3,4] model the signal power of wireless networks in complex environments with relatively low errors. Such systems use computer-based modeling tools to express the geometry of the region being studied. However, these models are simplifications of the real-world. When faced with the greater complexity of a real embedded environment, protocols encounter unforeseen edge cases and their performance suffers [5].

We take a different approach to modeling low-power wireless networks. Rather than simulate an arbitrary configuration of nodes based on analytical models, we examine how to simulate a specific configuration based on experimental traces. This empirical approach, which we call *physically-based simulation*, has an additional benefit – it allows us to validate our models by comparing simulation results to real-world measurements. Unlike purely analytical approaches, which are not grounded in a real network and therefore cannot be validated, using measurements from a deployment allows us to compare the resulting simulated behavior with the observed behavior.

Using physical-layer measurements in the form of variable frequency RSSI traces and 100 Hz or 1 kHz noise plus interference traces, we use probabilistic models to recreate behavior that is representative of what is observed on the real network. Simply replaying traces is insufficient, as it does not allow users to simulate experiments longer than the traces, and can also lead to overfitting to the particular trace used in the simulation [6].

Physically-based simulation has been applied successfully in our previous work to modeling noise and interference [7] for the purpose of simulation. In particular, we proposed the CLOSEST-FIT PATTERN MATCHING (CPM) algorithm for modeling noise and interference based on traces from deployed networks. In this paper, we propose extending this approach to modeling signal strength variations. CPM uses conditional probability distributions to model trends in the data trace. Based on the past k values for the variable (noise, signal strength, etc.), CPM samples from the probability distribution of what the next value will be. We present the algorithm in greater depth in Section 2.4.

There are two research challenges in applying this technique to simulating signal strength. The first is biased sampling. Unlike noise plus interference, which can be sampled at any time, signal strength can only be sampled on successfully received packets. This means that the signal strength trace is only partially observable. Only using received packet signal strengths skews the distribution and may lead to different packet reception ratios than those observed in reality. Therefore, an algorithm needs to generate estimates of missed signal strength measurements.

We propose two solutions to address biased sampling. One algorithm, called AVERAGE VALUE (AV), simply assumes all missed packets have the average observed signal power. In the other algorithm, we fill a probability mass function (PMF) of expected signal strengths based on the reception probabilities of the signal strengths of observed packets. We call this algorithm EXPECTED VALUE PMF (EVP). We find that in our experiments, EVP leads to a lower maximum packet reception ratio (PRR) error bound as compared to AV. EVP bounds the absolute error in PRR at 0.22 as compared to experiment, while AV bounds the same error measure at 0.3.

The second challenge is phase and sampling precision. For some physical layers, such as the one we study (IEEE 802.15.4), there is a very sharp 1.5 dB transition between low and high packet reception ratios. The radio hardware, however, can only produce readings at the precision of a single dBm. When the radio reports the signal strength of a received packet (RSSI, received signal strength indication), this is the sum of the noise plus interference and the actual signal strength. The sharpness of the SNR–PRR curve means that the signal strength reading must be precise and thus the relative phase of interference and signal is important.

To address the problem of phase and sampling precision, we explore whether assumptions of in phase, out of phase, or neutral phase additive models lead to more accurate simulation. We find that for each experiment we need to individually evaluate which phase assumption to use, and that choosing the correct assumption can lead to reductions of error in absolute packet reception rate by up to 0.3 in our experiments.

Another advantage of this model is that it allows network designers to choose a particular algorithm and phase assumption that best fits their deployment location. We provide an overview and several examples of fitting models to experimental traces in Section 5.1.

In addition, we compare fixed-PRR links of the various simulation methods to the Kantorovich–Wasserstein (KW) distance [8] on conditional packet delivery functions (CPDFs) [7]. CPDFs describe packet delivery probability given x consecutive successes or failures. As each value is equally important in a KW distance measure, CPDFs lend more weight to the rare than the common case, and so better represent the complexities of real-world networks than simple measures such as μ of a Gilbert–Elliot channel. The proposed model leads to a substantial reduction in the KW distance between CPDFs of bursts of receptions and losses as compared to CPDFs generated using a real signal power trace.

To verify the robustness of the methods to different sampling rates in experiments, we consider a large number

of experiments and evaluate the results using the same methods. We show that these simulation techniques are effective at different sampling rates and do not require a learning deployment at the same rate that will be used in the application network. Finally, we show that applying our model leads to a significant reduction in the factor-of-error in the number of parent changes in the standard TinyOS collection layer (CTP) in a simple test scenario as compared to the static signal power assumption that is commonly used in simulating low-power wireless networks.

The rest of this paper is organized as follows. In Section 2, we review related work about TOSSIM, a standard simulator for wireless sensor networks, about modeling signal strength, and about physically-based simulation. In Section 3, we present algorithms for the modeling of signal power in low-power wireless networks. In Section 4, we present our experimental work and provide an overview of the traces used to validate the proposed methods. In Section 5, we compare simulation results to experimental traces by using absolute PRR differences and the Kantorovich–Wasserstein (KW) distance using the concept of CPDFs. In Section 6, we consider varying the parameters of our model in terms of time measurement scales and perform an additional, comprehensive evaluation of our models. In Section 7, we show how the presented simulation model can allow simulators to more accurately capture the behavior of higher-level protocols. Finally, in Section 8 we provide concluding observations.

2. Background and related work

2.1. The TOSSIM Simulator

TOSSIM simulates TinyOS-based sensor network applications [9–11] using the IEEE 802.15.4 physical-layer. TOSSIM replaces several low-level hardware components with software equivalents. Application code runs unmodified in the simulator, enabling developers to test implementations in addition to algorithms. TOSSIM has advanced wireless network simulation features: it simulates packet capture, implements acknowledgments, including false positive acknowledgments, and has a robust noise model [7]. We use TOSSIM as the framework in which we implement and test our algorithms.

2.2. Signal power models for low-power wireless networks

Low-power wireless network simulators have taken varying approaches to modeling signal power. Released versions of TOSSIM, for example, assume signal power magnitude $|\mathbf{S}|$ to be constant, and allow the user to input a gain (attenuation) value for each link in the simulation. Currently, gain is either manually input for each link or modeled using a tool [12] which simulates overall network structure, but not the temporal variations in individual links in the network. Fig. 1 illustrates the noise plus interference and RSSI variations over typical, representative experiments. There is a longer term variation in the signal power of received packets, which is approximated in this figure by $\text{RSSI} = |\mathbf{S} + \mathbf{N}|$, where $|\mathbf{N}|$ is the noise plus interference magnitude. Since $|\mathbf{S}| \gg |\mathbf{N}|$ for received packets, however, it is unlikely that the variations illustrated in this figure, taken from experimental traces, are a result of noise variations alone. Thus, TOSSIM's assumption that signal power is constant is a simplification to reality.

Other models of low-power wireless networks have attempted to apply the aforementioned analytical approach. One such analytical model that has been investigated in the context of low-power sensor networks is the combined *simplified path loss* model and *log-normal shadowing* random process [13]. On the dBm scale, the signal power S is given by

$$S\ \text{dBm} = P_t\ \text{dBm} + K\ \text{dB} - 10\lambda \log_{10}\frac{d}{d_0} + \Psi_{\text{dB}}(\mu, \sigma), \qquad (1)$$

where S is the desired signal power, P_t is the transmit power, K is a unitless constant, λ is the path loss exponent, $\frac{d}{d_0}$ is a physical parameter proportional to distance, and Ψ is a Gaussian random variable, where μ is the mean and σ is the standard deviation. Bae and Kim [14] conducted an investigation applying this model for use with the TI/Chipcon CC2420, a radio commonly used in sensor network deployments, including the experiments we performed, and gave a set of parameters that apply to this radio and to the 2.4 GHz frequency range. When we apply this model, we use the parameters given in Table I of that study [14].

2.3. Physically-based simulation

This paper extends the physically-based radio link simulation algorithm introduced in our previous work [7]. The equation that underlies this model is

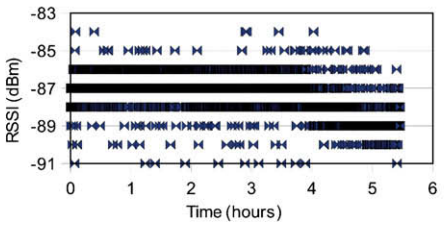
$$\text{SNR} = \frac{|\mathbf{S}|}{|\mathbf{N}|}, \qquad (2)$$

where $|\mathbf{S}|$ is the magnitude of the signal power of a received packet, $|\mathbf{N}|$ is the resultant magnitude of any environmental noise or disruption not caused by the network being studied, and SNR is the signal-to-noise ratio [7]. On a logarithmic scale, expression (2) may be expressed as

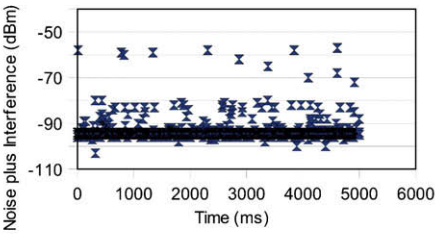
$$\text{SNR}_{\text{dB}} = |\mathbf{S}|_{\text{dBm}} - |\mathbf{N}|_{\text{dBm}}. \qquad (3)$$

This expression for SNR can be mapped to a packet reception rate using the function given in Fig. 2. The derivation of this function assumes the probabilistic independence of several random variables [12,15] and has been verified experimentally. We maintain these assumptions for the simplicity of analysis, and we see their relaxation as important future work to pursue to make models such as the one proposed here even more accurate.

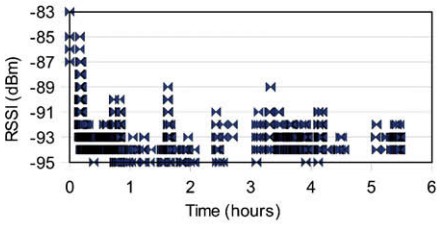
The TOSSIM simulator implements this model with a variable noise parameter: it models $|\mathbf{N}|$ using the CPM algorithm [7], reviewed in the next section. We see from Fig. 1 that modeling noise is desirable since there are substantial short term variations in noise values. As mentioned above, released versions of TOSSIM assume $|\mathbf{S}|$ to be constant.



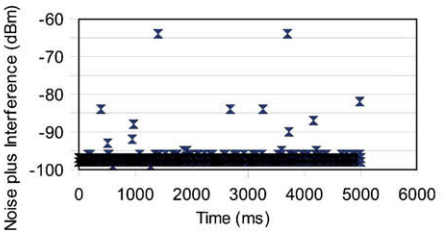
(a) RSSI variation for experiment with PRR = 0.59.



(b) Noise plus interference variation of experiment with PRR = 0.59.



(c) RSSI Variation for experiment with PRR = .08.



(d) Noise plus interference variation of experiment with PRR = .08.

Fig. 1. This figure shows experimental variations in RSSI ((a) and (c)) and noise plus interference values ((b) and (d)) from two representative sensor network deployments. We see that both of these parameters are not constant and do not vary consistently across different environments.

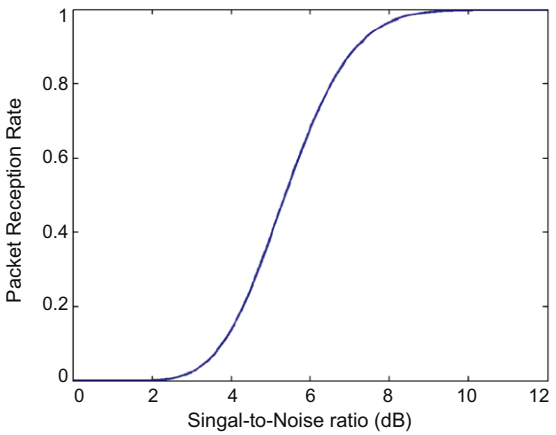


Fig. 2. TI/Chipcon CC2420 SNR/PRR curve [10].

2.4. The Closest-Fit Pattern Matching (CPM) Algorithm

The CPM algorithm uses an experimental trace to create a conditional model of observed values [7]. First, an experimental learning trace is collected at frequency f in the environment to be simulated. In its model generation phase, CPM scans the trace and computes a probability distribution of the next value v given k prior values in order. To run CPM, a simulation replays the first k values from the trace; then the algorithm uses the probability distribution constructed during model generation to sample the next value. If the prior k values do not match a pattern observed in the real network, then CPM samples from the most common pattern.

If k is equal to the length of the trace, then CPM will simply replay the trace. If $k = 0$, then CPM takes independent samples. Our previous work [7] found that $k = 20$ leads to the best results when simulating noise plus interference, and we use this value of k unless otherwise noted.

3. Algorithms for modeling signal power

We propose to collect signal power traces over long periods of time and then to use the CPM algorithm to predict signal power. We consider two algorithms to fill in lossy samples of signal power from experiments – the EXPECTED VALUE PMF (EVP) and the AVERAGE SIGNAL POWER VALUE (AV) algorithms.

3.1. Collecting signal power traces

Collecting signal power traces is more intricate than collecting noise traces for several reasons. First, any signal power value refers to a link between a pair of nodes; thus, any experimental trace needs to involve both a sender and a receiver. Noise, on the other hand, is local to an individual node in the network.

Furthermore, signal power is not known to the mote directly. The best estimate of its magnitude is the RSSI $= |\mathbf{S} + \mathbf{N}|$ upon packet reception. Due to the steepness of the SNR–PRR curve (Fig. 2), RSSI must be corrected for this noise error to obtain the correct signal power magnitude. This is not just a matter of subtracting the noise value since the phases of waves must be considered.

In addition, while noise can be sampled in any environment, and there will always be a sample when one is

requested, this is not the case for signal power. Recall that signal power requires the communication between two nodes; if the link fails and a packet is not delivered, then there will be no RSSI or signal power value available. Thus, the signal power trace needs to be post-processed and filled-in for completeness and to avoid missing samples.

We suggest two separate steps to account for these challenges: (1) filling in missing signal power values into the experimental trace, and (2) correcting for the phase differences between noise and signal traces. Each of these algorithms assumes that a trace of RSSI values has been collected by measuring the received signal power from a network of two motes. This is accomplished by a TinyOS application that sends packets at the constant frequency f from a designated sender mote, and records the RSSI value at reception; we denote this RSSI trace (measured in units of dBm) $\mathscr{R}$. The algorithms also require that the noise plus interference in the environment that we aim to simulate has been characterized, such that the average noise N (measured in units of dBm) over the period of the RSSI-collection experiment is known or can be approximated well. A TinyOS application similar to `RSSISample` [16] can be used to measure average noise.

3.2. Phase differences

Furthermore, we need to determine the phase differences between noise and signal, which we denote p. In principle, p could be anywhere in the continuous range between -1 and 1, since it is the phase difference between the resultant noise vector and the signal power vector, aggregated over the total time of the experiment or simulation. However, as we show below, we can reduce the continuous search space of possible phase differences and derive satisfactory simulation results by making one of three simple assumptions about p, where

$$p = \begin{cases} -1, & \text{Noise and signal are assumed in phase} \\ 0, & \text{No correction for phase difference} \\ 1, & \text{Noise and signal are assumed out of phase} \end{cases}. \tag{4}$$

An interesting and important question is how to select the correct phase assumption for a given scenario. In course of extensive experimentation, including many experiments that are not explicitly presented in this paper, we found that the out-of-phase assumption ($p = 1$) is the most effective in the largest number of cases. However, this difference may not be statistically significant. It may be possible to determine p directly by using a spectrum analyzer. Such analysis would need to account for all of the noise sources, their time variance, and to aggregate p over the entire length of the experiment. Additional work is needed to determine p in a simple and direct way. We explore related problems in separate work, suggesting a parsimonious understanding of the wireless channel [17,18]. In this paper, we test all three assumptions given above by setting the phase correction factor p when constructing the signal power trace from the RSSI trace.

Algorithm 1 (*EXPECTED VALUE PMF*).

Data : Average noise N (in dBm), list of successful reception times $\mathscr{T} \subseteq [0, \ell]$, RSSI trace $\mathscr{R} : \mathscr{T} \rightarrow$ dBm value collected from experiment of length ℓ at frequency f, and phase assumption p

Result : A filled in signal power trace $\mathscr{S} : [0, \ell] \rightarrow$ dBm value mapping for time duration ℓ at frequency f

Initialize PMF $\mathscr{P}$;

foreach $t \in \mathscr{T}$ **do**

  $r = \mathscr{R}(t)$;

  Find $s = 10\log_{10}\left(10^{\frac{r}{10}} + p \times 10^{\frac{N}{10}}\right)$;

  Add mapping $(t,\ \text{round}(s))$ to $\mathscr{S}$;

  Add s to $\mathscr{P}$ with frequency $\frac{1}{prr(s-N)} - 1$;

end

foreach $(t \in [0, \ell]) \notin \mathscr{T}$ **do**

  Add mapping $(t,\ \text{sample}(\mathscr{P}))$ to $\mathscr{S}$;

return $\mathscr{S}$;

3.3. THE EXPECTED VALUE PMF (EVP) algorithm

First we consider the EXPECTED VALUE PMF (EVP) algorithm. Pseudocode is given in Algorithm 1. The algorithm accepts as input the average noise in the environment to be simulated N (in dBm), a set of successful reception times $\mathscr{T}$ from an experiment conducted at frequency f, an RSSI trace $\mathscr{R}$ which maps time values where the RSSI is known to the corresponding RSSI values measured in dBm, and a phase assumption p. Then, for each of the known RSSI values, a signal power value is generated by correcting the RSSI value for the noise error using the expression

$$s(t)\ (\text{in dBm}) = 10\log_{10}\left(10^{\frac{\mathscr{R}(t)}{10}} + p \times 10^{\frac{N}{10}}\right). \tag{5}$$

The intuition is that if a signal and noise are in phase, then the actual signal power is lower than the RSSI value detected, so $p = -1$. If the signal and noise are out of phase, then the actual signal power is higher than the RSSI value detected, so $p = +1$. Finally, if the phase differences cancel each other out, then $p = 0$ and $s(t) = \mathscr{R}(t)$. After it is computed, $s(t)$ is added to the signal power trace $\mathscr{S}$ at time t.

The algorithm also adds each signal power $s(t)$ computed from the experimental trace to PMF $\mathscr{P}$ at a frequency corresponding to the number of packets that are expected to be lost, quantified by

$$\text{Expected lost packets} = \frac{1}{prr(s-N)} - 1 \tag{6}$$

where $prr(\cdot)$ maps a signal to noise ratio (SNR $= s - N$ on the dB scale) to the corresponding PRR value, following the curve illustrated in Fig. 2. This expression effectively extrapolates the number of packets that should have been received at this signal power value, based on the probability of receiving this single packet. One is subtracted to account for the packet that has just been received. This number is stored as a `float` value in our algorithm, so it is possible to have fractional amounts of expected missing packets.

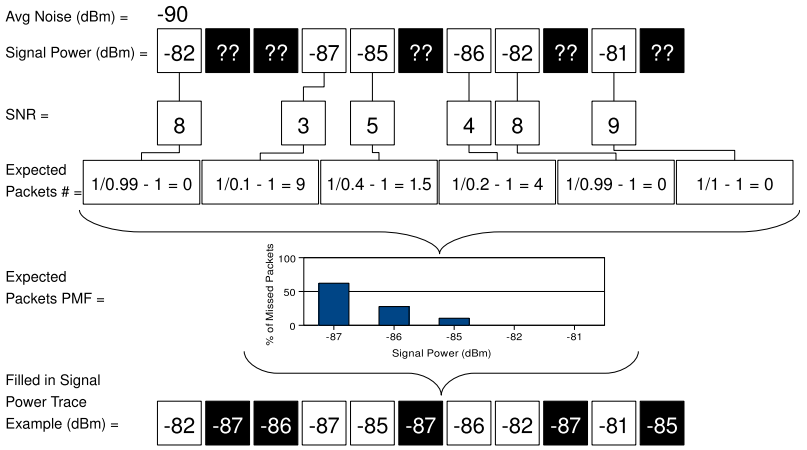


Fig. 3. An example of the expected value PMF algorithm on an input, assuming an average noise of −90 dBm. This trace shows 6 of 11 packets were received, with RSSI values of −82, −87, −85, −86, −82, and −81 dBm. The RSSI values of the five missed packets are not known, and this is indicated by "??"s in the figure. Extrapolating from expected PRRs of the RSSI values of the received packets, there should be 14.5, not 5 lost packets: for example, only 1 of 10 packets at −87 dBm should be received. Note that for simplicity, we use PRR values that are approximates to those given in Fig. 2 and that we round the expected number of packets to one decimal point and omit trailing zeros.

Finally, for every missing signal power value in $\mathscr{S}$, i.e., for every time that a packet was lost in experiment, the EVP algorithm samples the PMF of expected missing values $\mathscr{P}$, and these times are mapped to signal power values corresponding to the proportion that each integral signal power value is found in the PMF generated from the experimental trace.

The number of expected lost packets is greater than the actual number of lost packets in the example of the EVP algorithm shown in Fig. 3. This is also the case in the real traces studied. We conjecture that this may be because of the 1 dBm granularity of the RSSI values measured by the CC2420 radio in our experiments. However, the full reason for this difference and its causes and implications is a topic of future work that we are interested in pursuing. As expected, this observation results in lower power values being common in traces filled in using the EVP algorithm since the very low delivery ratio of these values skew the distribution.

3.4. The AVERAGE SIGNAL POWER VALUE (AV) algorithm

We also consider the AVERAGE SIGNAL POWER VALUE (AV) algorithm. Pseudocode is given in Algorithm 2. This algorithm accepts as input the average noise N (in dBm), a set of successful reception times $\mathscr{T}$ from an experiment conducted at frequency f, an RSSI trace $\mathscr{R}$ which maps time values where the RSSI is known to RSSI values measured in dBm, and a phase assumption p. Then, the signal power trace is computed for these known times using expression (5) and all known values are added to the signal power trace $\mathscr{S}$. The average signal power of the received packets, P, is computed (in units of dBm) and rounded to an integer, to conform to the output of the CC2420 radio for RSSI values. This average signal power has a sampling bias, as it only considers received packets. Finally, the algorithm fills in the signal power trace by inserting the average signal power value for all time values that are missing from $\mathscr{S}$. Fig. 4 shows an example of the execution of the AV algorithm.

Algorithm 2 (*AVERAGE SIGNAL POWER VALUE*).

Data : Average noise N (in dBm), list of successful reception times $\mathscr{T} \subseteq [0, \ell]$, RSSI trace $\mathscr{R} : \mathscr{T} \rightarrow$ dBm value collected from experiment of length ℓ at frequency f, and phase assumption p
Result : A filled in signal power trace $\mathscr{S} : [0, \ell] \rightarrow$ dBm value mapping for time duration ℓ at frequency f
for each $t \in \mathscr{T}$ **do**
  $r = \mathscr{R}(t)$;
  Find $s = 10\log_{10}\left(10^{\frac{r}{10}} + p \times 10^{\frac{N}{10}}\right)$;
  Add mapping (t, s) to $\mathscr{S}$;
end
Let P be the average value of power values in $\mathscr{S}$ (in dBm), rounded to an integer;
for each $(t \in [0, \ell]) \notin \mathscr{T}$ **do**
  Add mapping (t, P) to $\mathscr{S}$;
return $\mathscr{S}$;

3.5. Implementation and performance of signal power correction algorithms

We implemented both algorithms as a Java preprocessor that accepts text files of traces that are collected from a TinyOS application we developed for this purpose. The application outputs the signal power traces $\mathscr{S}$ to a text file that is used as input to a modified version of the TOSSIM simulator, which uses CPM (see Section 2.4) to predict signal power when a simulation is performed. Our preprocessor also automatically generates Python scripts for use with TOSSIM for all three phase assumptions discussed above. The preprocessing step needs to be run only once for an arbitrary number of simulations of a certain wireless link.

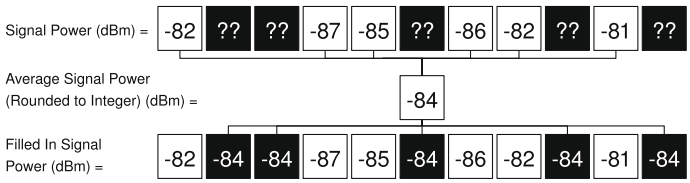


Fig. 4. An example of the average signal power value algorithm. This trace shows 6 of 11 packets were received, with RSSI values of −82, −87, −85, −86, −82, and −81 dBm. The RSSI values of the five missed packets are not known, and this is indicated by "??"s in the figure. In this algorithm, these missing values are filled in with the average signal power value, rounded to an integer.

We measured the running time of the preprocessing code using the suggested algorithms on a laptop with a 2.6 GHz processor. This run generated signal power traces and TOSSIM scripts for both directions of the link under the three phase assumptions studied. The RSSI traces had over 80,000 samples before they were filled in. Times were measured using Java's `System.currentTimeMillis()` function. The EVP algorithm ran in 11.529 s, while the AV algorithm ran in 11.244 s.

To decide on the effectiveness of these simulation techniques, we collected experimental noise and signal power traces discussed in the next section.

4. Data collection methodology

In conducting this study, we noticed a lack of experimental traces for signal power variations. Thus, we collected our own experimental traces in order to validate simulations using the proposed algorithms. We conducted numerous experiments that have one sender and one receiver, placed at specific locations on the Cornell University campus shown in Fig. 5. In addition, we conducted a high-frequency experiment at a testbed in Gates Building at Stanford University, again using one sender and one receiver.

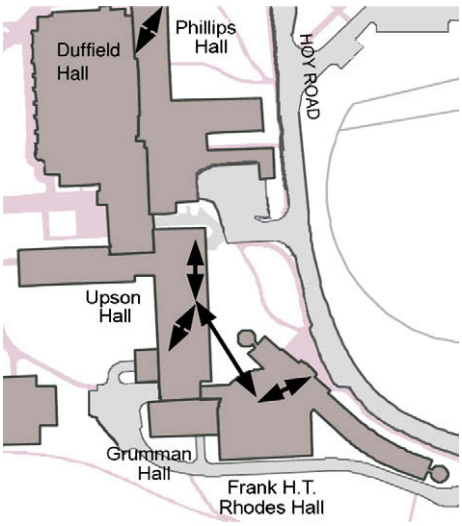


Fig. 5. Map of experimental collection locations at Cornell for this investigation; each pair is represented by an arrow. The link in Phillips hall had two motes separated by one floor, and in all other experiments the motes were located on the same floor. Base map from http://www.parking.cornell.edu/pdf/Stu_combo_2006.pdf.

We developed a TinyOS application that sends packets at various frequencies f from the sender to the receiver. This application does not implement a particular protocol to guarantee packet delivery – there are no acknowledgements and no retransmissions, for example. In fact, we are interested in studying the reception patterns of packets and the signal power of such received packets. We chose $f = 4$ Hz as a baseline collection frequency to investigate long RSSI traces for the Cornell campus experiments. The Gates Building experiment, used to investigate the influence of collection frequency on our algorithms, was collected at $f = 100$ Hz and then subsampled to simulate collection at a variety of lower frequencies. The sender mote sends packets at these rates; the receiver simply listens for packets and records the sequence number and the RSSI of each received packet. For each experiment, at Cornell we tested the link in both directions, i.e., first one mote is the sender and the other mote is the receiver, and then the sender and the receiver are switched for the purposes of collecting another trace. We collected a trace of about 12 h (or more) for each pair of nodes at Cornell, and for about three hours in the Gates Building experiment.

The nodes used were TelosB motes [19] with TI/Chipcon CC2420 radios [20], the radio that TOSSIM models. The Cornell campus experiments were conducted in Rhodes, Upson, and Phillips Halls. These buildings are high traffic, high use academic facilities around the clock and they have a pervasive 802.11 abg wireless network. There are also cordless phones, microwaves, and personal wireless access points in use in both buildings. Gates Building has an 802.11 g wireless network, and there are microwaves and other interference sources in use as well. Thus, in both environments, there are many factors that can impact the quality of the wireless connection between the nodes. While both are academic environments, the building age, composition, and location varies widely among the different experimental traces used in this study.

5. Evaluation

We evaluate the proposed model on two levels. First, we measure the error in packet reception rates by comparing simulation traces to the corresponding experimental traces. We show that with an appropriate choice of assumptions about phase differences and the filling-in algorithm between the signal and noise, we can achieve predictive packet reception rate simulation. Then, we eval-

uate the correlation among packets and show that our algorithms provide KW distances lower by a factor of about three compared to the best alternative simulation methods studied.

5.1. Comparing simulation and experiment PRRs

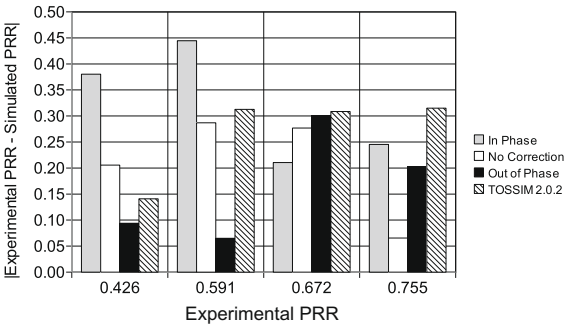
In this section, we compare a first-order parameter, packet reception rate (PRR), between simulation and the environment that we are trying to simulate. The different algorithms proposed above have different levels of correspondence to experiment with respect to PRR.

PRR is a very basic simulation parameter. It is possible to get a perfect PRR simulation by simply accepting packets at a rate equal to the PRR that was derived from the experiment. Unfortunately, such a simulation will not take into account temporal variation of signal power or packet reception correlation that is known to be found in low-power wireless links [7,17,18,21]. Furthermore, in complex networks it may be impossible to dictate the reception rate, since such a simulation does not consider the interactions between different pairs of nodes that may transmit concurrently.

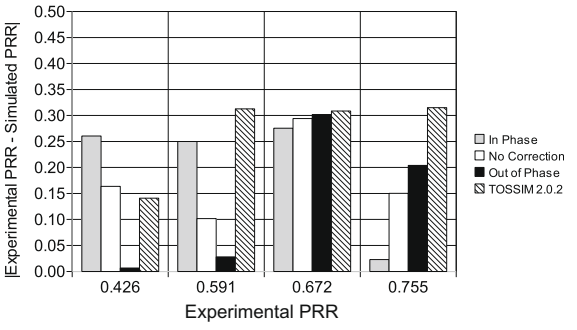
PRR is a very difficult parameter for general simulators to get right. For example, we ran simulations of our experiments with the analytical model discussed in Section 2.2 and got PRRs that were very different from those that we observed experimentally. In most cases, these results were so far off that they do not even provide a basis for comparison.

At the same time, correctly modeling the packet reception rate is extremely important in wireless networks, especially for those links in the intermediate range. It is vital for protocol designers to have an idea of the proportion of packets that are received as compared to those that are lost in order to correctly account for these losses, either in information, time, or both, when programming the network and designing applications. Physically-based simulation introduced in TOSSIM 2.0.1 and 2.0.2 has greatly increased the ability of the simulator to correctly capture packet reception rates. For example, Metcalf [22] shows that TOSSIM 2.0.2 produces largely correct results in terms of PRR for good and bad links. In that work, TOSSIM's gain value is set to the average RSSI of the link, which approximates signal power. However, examining the figures in Chapter 4 of [22], we see that PRR differences between experiment and simulation values are greater in intermediate links. Although there are relatively few intermediate links in that study, their PRR is not predicted precisely in many cases for such link qualities.

The model that we propose improves upon the prediction of PRR for the following reasons. First, it considers the variations in signal power which may account for PRR variations. There are two algorithms proposed for filling-in experimentally determined signal power traces, and due to varying environmental conditions one or the other may be more appropriate. In addition, our model corrects for the phase differences between signal and noise waves for each of the links when converting RSSI to signal power. This correction needs to be tuned to the environment, and it can cause major differences in the overall PRR of a given link.



(a) Expected Value PMF (EVP) Algorithm.



(b) Average Value (AV) Algorithm.

Fig. 6. Absolute differences between the PRR of experiments and simulations for various assumptions about phase using (a) the EVP algorithm and (b) the AV algorithm for filling in signal power traces. In both plots, the TOSSIM 2.0.2 value takes gain (signal power) to be the average RSSI, without rounding.

Note that in any physically-based simulation model, it is almost always possible to get an exact PRR by modifying the signal power with a certain coefficient or an additive value. This is, essentially, a brute-force method of searching for the appropriate phase correction factor p. Our proposed method shows that using a signal power trace with just three assumptions about phase, it is usually possible to achieve nearly the same result.

In Fig. 6, we compare the absolute differences between experimental and simulation PRRs of the various algorithms that we propose in this work under different assumptions about phase. We also perform the same comparison to the corresponding experiments as simulated by TOSSIM 2.0.2. The TOSSIM simulator assumes that signal power is constant, and we input the average RSSI value of the corresponding experimental trace into TOSSIM. This is a common assumption; for example, Metcalf used this approximation as the gain parameter in TOSSIM in the aforementioned study [22]. In each case, noise plus interference is modeled by CPM using a noise trace collected in the experimental environment without sending packets.

In the present experimental study, we noted similar results – the bad links and the good links perform sufficiently well in TOSSIM 2.0.2. As we can see in Fig. 6, however, TOSSIM does not give satisfying results for intermediate links and gives a somewhat arbitrary PRR error given a consistently calculated gain value.

By correctly tuning the choice of assumptions it is possible to effectively simulate a wireless link using the suggested technique. Given the appropriate phase assumption, the EVP algorithm approximates PRR to within an absolute difference of 0.22 in the worse case (less than 0.1 in all but one case), while the AV algorithm approximates the PRR value to within 0.3 (less than 0.1 in all but one case).

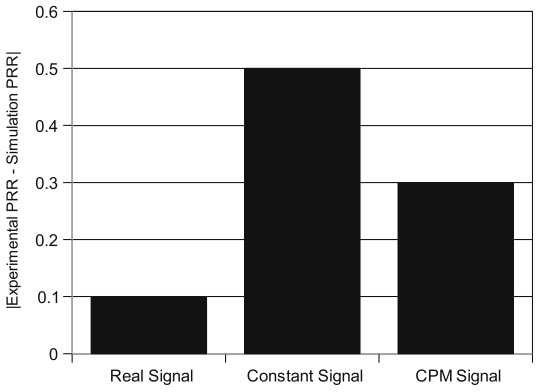


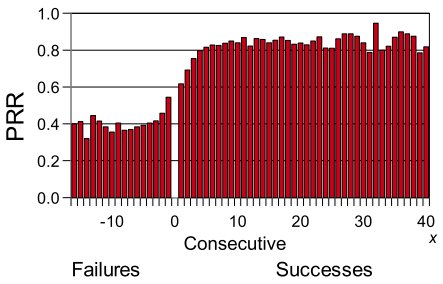
Fig. 7. Absolute differences between the PRR of experiments and simulations for a single experiment with a PRR of about 0.6, using the CPM method to simulate environmental interference plus noise and various strategies for modeling signal power.

Conducting an analysis similar to Fig. 6 allows network designers to fit experimental data from their deployment location to one of the algorithms proposed. Thus, it is possible to use this method to choose the best simulation technique for a given wireless link.

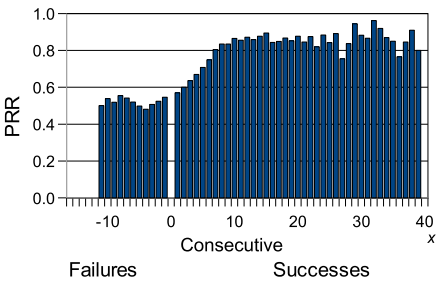
In Fig. 7, we analyze a particular experiment with a packet reception rate of 0.6. We then plot the error in simulation using real signal power, the CPM model with the out of phase assumption and the EVP filling-in algorithm, and the PRR with a constant, average signal power. First, this figure provides validation for our physically-based simulation model, as the error using real signal power is 0.1. While the error in the constant signal power assumption is 0.5 (used currently in TOSSIM), the error using the proposed algorithm is 0.3, an improvement that may be important in some applications.

5.2. KW distances of fixed-PRR simulations

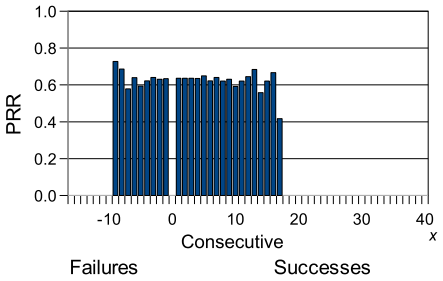
Not only is our approach effective at more correctly modeling experimental PRRs in simulation, it can also model packet reception correlation effectively. Such correlations were shown to have an effect on higher-order protocols [7] and we provide additional evidence of this in Section 7. Clearly, the overall PRR of a long-term experiment is not the only factor that has to be considered for correct simulation. We also need the ability to appropri-



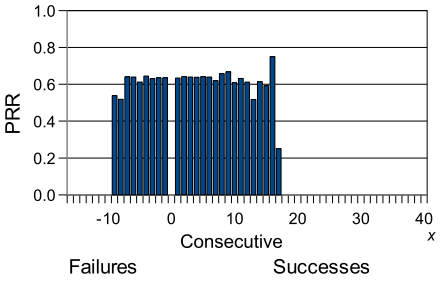
(a) Real signal power.



(b) CPM signal power. KW distance compared to real signal power = 0.06.



(c) Constant signal power. KW distance compared to real signal power = 0.21.



(d) Combined path loss and log-normal shadowing signal. KW distance compared to real signal power = 0.20.

Fig. 8. CPDFs for PRR = 0.59 link. The negative horizontal axis plots consecutive packet failures, while the positive horizontal axis plots consecutive packet receptions. We note that the real signal power plot (a) is generally correlated in *x*, plotted on the horizontal axis, with a sharp increasing trend as *x* increases, and that the trace simulated with CPM (b) also follows such a correlation. The constant signal CPDF (c) and the combined path loss and log-normal shadowing simulation CPDF (d) show no apparent correlation. Values with no bar indicate that no data was collected for this value, not a PRR of 0.

ately account for temporal variations such as burstiness in packet reception that are of fundamental importance in designing effective protocols.

To investigate the impact of applying our model for signal power on packet reception correlation, we apply a constant noise so that signal power is the only factor varied. Then, we change the noise value for the various simulation models such that the packet reception rates are essentially the same. We consider the following several signal power modeling algorithms in this fashion. *Real signal power* traces are from actual experiments and are corrected and filled-in using the algorithms given above. *CPM signal power* traces are generated by inputting the real power traces into the CPM algorithm, keeping a history of $k = 20$, and generating a simulation trace. *Constant signal power* traces assume that signal power is constant and pre-determined. *Combined path loss and log-normal shadowing* signal traces use the analytical expression (1) suggested in Section 2.2 to model signal power.

For each of these simulations, we build a conditional probability plot of packet reception based on the condition

$$\mathscr{X}(x) = \begin{cases} |x| \text{ sequential packet losses}, & x < 0 \\ x \text{ successful sequential receptions}, & x > 0 \end{cases}, \quad (7)$$

following an idea first introduced in our previous work [7]. Such a distribution has been called a CPDF. CPDFs investigate trends in packet reception burstiness [17,18,21] on the time-scale of one packet only.

In Figs. 8 and Figs. 9, we present CPDFs of two intermediate links that we investigated. The experiment from which this link was collected received about 0.59 of the packets, and varying the constant noise value led to the two different PRRs for the links in the two figures. For the purpose of this analysis, we used the EVP algorithm, which provides the tightest maximum error bound in PRR over many experiments. In addition, we use the out of phase assumption, which provides a PRR close to the

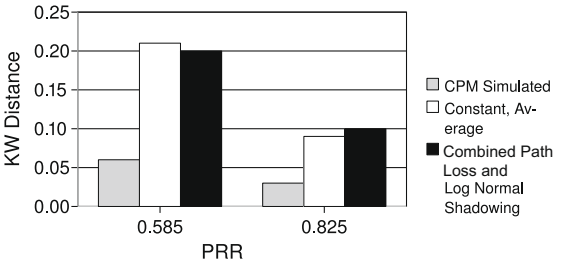
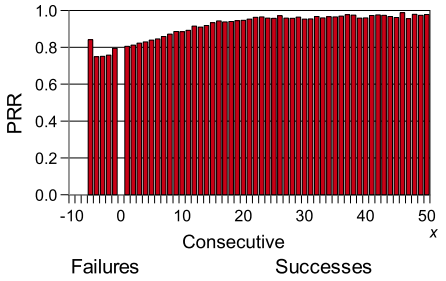
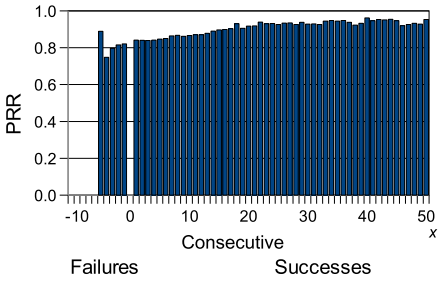


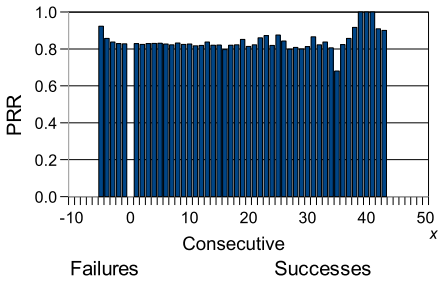
Fig. 10. The KW distance comparing CPDFs of the various signal power modeling algorithms to CPDFs from a real signal power trace. The KW distance of the simulation using the CPM algorithm is substantially lower than the KW distances corresponding to other simulation techniques.



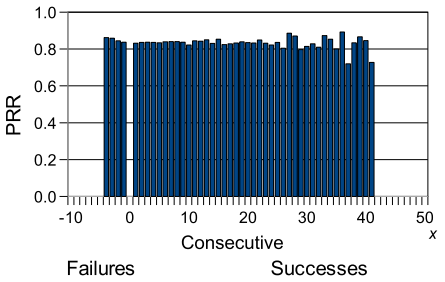
(a) Real signal power.



(b) CPM signal power. KW distance compared to real signal power = 0.03.



(c) Constant signal power. KW distance compared to real signal power = 0.09.



(d) Combined path loss and log-normal shadowing signal. KW distance compared to real signal power = 0.10.

Fig. 9. CPDFs for PRR = 0.83 link. The negative horizontal axis plots consecutive packet failures, while the positive horizontal axis plots consecutive packet receptions. We note that the real signal power plot (a) is generally correlated in x, plotted on the horizontal axis, with a gently increasing trend as x increases, and that the trace simulated with CPM (b) also follows such a correlation. The constant signal CPDF (c) shows no apparent correlation, while the combined path loss and log-normal shadowing simulation CPDF (d) appears to have a slightly decreasing PRR value as x increases. Values with no bar indicate that no data was collected for this value, not a PRR of 0.

experimentally discovered PRR using the real experimental noise. Note that the CPDF corresponding to the CPM algorithm shows a more similar correlation to the real signal CPDF than any other simulation method. In Fig. 8a, we note that there is an overall sharp increase in PRR as x (represented on the horizontal axis of the CPDFs and defined in expression (7)) increases. The CPDF corresponding to the CPM model (Fig. 8b) is the only other that appears to show any such increase. On the other hand, the CPDFs of constant and combined path loss and log-normal shadowing signal simulations (Figs. 8c and d) appear to present little correlation. Similarly, in Fig. 9a, we see a more gentle increase in PRR as x increases. Again, the CPM signal CPDF (Fig. 9b) is the only simulation model that captures this increase effectively. The constant signal CPDF (Fig. 9c) shows no apparent correlation and the CPDF generated from the combined path loss and log-normal shadowing model (Fig. 9d) actually shows a slightly decreasing PRR as x increases. Recall that CPDFs are generated from a probabilistic simulation, so some outliers in the overall trends may occur.

To quantify these observations, we measure the Kantorovich–Wasserstein (KW) distance [8] between the overlapping portions of CPDFs of real signal power simulations and CPDFs of the simulation techniques discussed above. The KW distance computes the distance between probability distributions in a manner that places more emphasis on the rare rather than the common situation. In Fig. 10, we show the results of this calculation. The code used to compute the KW distance uses an equivalent metric known as Earth Mover's distance [23]. As expected, the KW distance between the CPM simulation CPDF and the real signal power CPDF is much lower than the KW distance observed when comparing the real signal power CPDF to the CPDF corresponding to any other simulation model.

In particular, the CPM algorithm improves the KW distance of the intermediate links studied by a factor of about three. This shows that applying CPM to the signal power traces output from the algorithms suggested in Section 3 is effective at modeling the temporal correlation of packets. At the same time, the global behavior of the link is similar to the behavior (i.e., the PRR) of the corresponding experimental link. Thus, applying our model leads to a radio link simulation that is quite similar to the behavior of the real link being modeled, both in terms of packet reception correlation and in terms of packet reception rates.

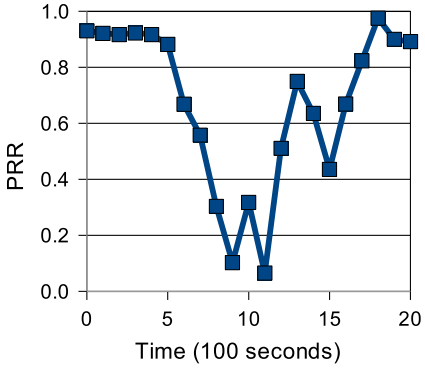
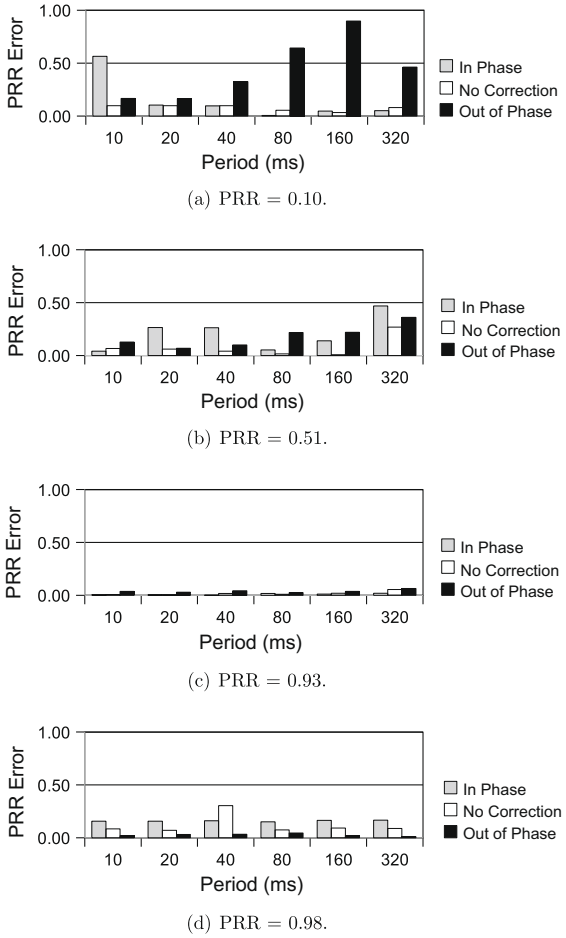


Fig. 11. PRRs of individual partitions of the Gates Building experiment used for evaluation at various time-scales, each marked with a box. Each of these partitions was 100 s in length, and all 21 partitions were used to evaluate the proposed simulation method. This trace was chosen due to the wide variety of packet reception rates represented in the individual partitions, ranging from 0.06 to 0.98 PRR. This figure is drawn with respect to the real temporal variation of this link.

6. Varying model parameters: time-scale considerations

6.1. Observations and methodology

Wireless links tend to be bursty in reception and at the physical-layer [7,17,18,21,24]. Here we study the impact of burstiness on the proposed models by varying model parameters to various time-scales. Our proposed algorithms consider time-scales in three model parameters: (1) the length of the underlying trace ℓ, (2) the value of k, the history parameter in the CPM algorithm, and (3) the collection frequency of the underlying signal power trace f.



(a) PRR = 0.10.

(b) PRR = 0.51.

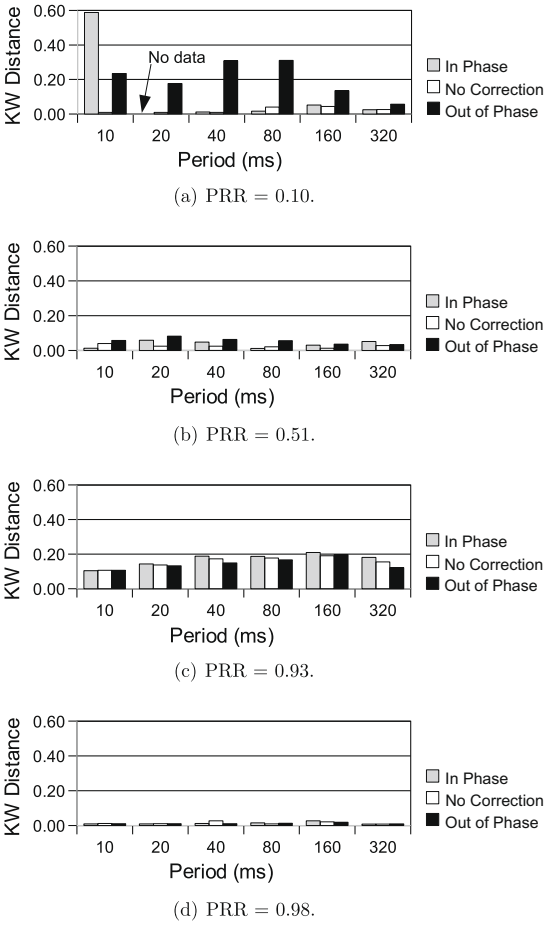
(c) PRR = 0.93.

(d) PRR = 0.98.

Fig. 12. PRR error (absolute value of the difference between experimental PRR and simulation PRR) at different sampling frequencies and with different phase assumptions.

To study the impact of burstiness on the underlying wireless channel, we use the high-frequency Gates Building trace. This trace includes a large number of packets collected at a 100 Hz frequency. To vary f, we subsample partitions of the Gates Building trace at a set interval. To avoid biasing the sample as much as possible, we hold both the value of $\ell \times \frac{1}{f}$ and the value of $k \times \frac{1}{f}$ constant, using the baseline $\ell \times \frac{1}{f} = 100,000$ and $k \times \frac{1}{f} = 5000$, corresponding to the values of k and f introduced in the previous section for evaluation. This allows us to study the impact of burstiness on the underlying models proposed in this paper. We then consider the error in packet reception rate and the KW distance between the original experimental trace and the corresponding simulation. Note that only the trace used for modeling is subsampled, while the trace used to evaluate the results is the original experimental trace at the full 100 Hz frequency. This analysis allows us to study the underlying effects of burstiness on our method for simulating wireless links.

We evaluate the various time-scales at $f = 100$, 50, 25, 12.5, 6.25, and 3.125 Hz packet sampling frequency. The corresponding periods, i.e., the time between successive packet transmissions (used in Figs. 12 and 13), are $\frac{1}{f} = 10$, 20, 40, 80, 160, and 320 ms, respectively. To avoid overfitting, we ran simulations at the full frequency of 100 Hz. We evaluated a total of 21 partitions (sections) of the Gates Building experiment using these methods. We chose this trace and these partitions because of the wide variety of packet reception rates represented in these partitions; see Fig. 11. These partitions were simulated using the EVP algorithm with each of the three phase assumptions proposed.



(a) PRR = 0.10.

(b) PRR = 0.51.

(c) PRR = 0.93.

(d) PRR = 0.98.

Fig. 13. KW distances between experimental and simulation traces, at different sampling frequencies and with different phase assumptions. No KW distance was calculated in one of the scenarios due to no packets being received in the simulation trace marked with "No Data".

6.2. Simulating signal power with traces at various time-scales

The simulation methods suggested in this paper are effective regardless of the experimental collection frequency at the initial deployment used for collecting signal power traces. Since the model is robust to changes in packet frequency, this means that the initial learning phase does not need to be at the precise frequency used by the application network and that the learning phase does not need to be repeated if the frequency of sending packets varies with differing applications of the same wireless link. This observation greatly increases the applicability of our model for use in the simulation of arbitrary, multipurpose low-power wireless networks.

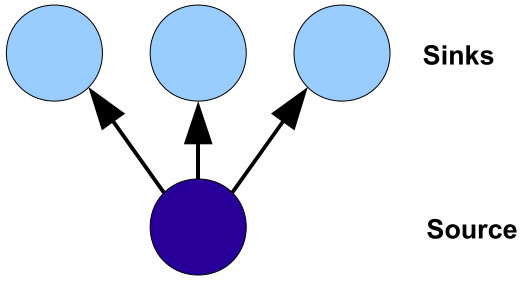


Fig. 14. Four-node topology, where one node is the packet source and the remaining three are the sinks. Packets are routed from the source to the sinks, as indicated by the arrows. These direct paths are connected using the Gates Building wireless link considered above. There are no other wireless links available for communication in this simulation.

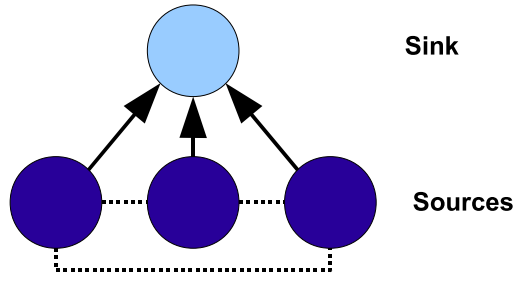
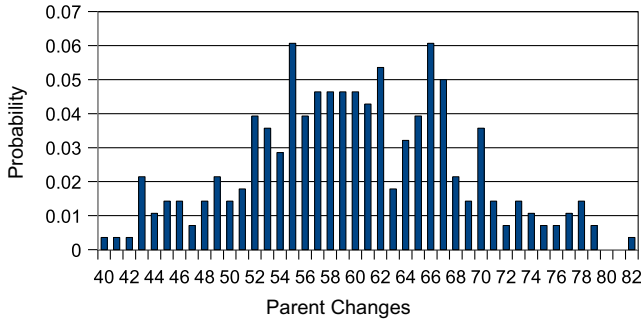


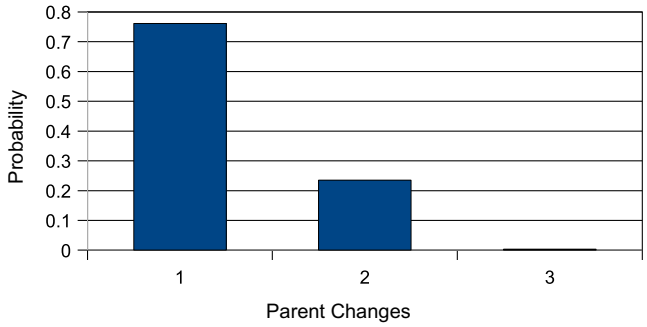
Fig. 15. Four-node topology, where three nodes are packet sources and the fourth is the sink. Packets are routed from the sources to the sink, as indicated by the arrows. These direct paths are connected using the Gates Building wireless link considered above. In addition to the direct paths, the sources are also connected by the Gates Building wireless link, as indicated by the dotted lines.

We use the same evaluation techniques as in Section 5 to verify the effectiveness of this simulation technique – absolute difference between PRR of experiment and simulation (PRR error) and the KW distance between CPDFs of our simulation model and real signal power traces.

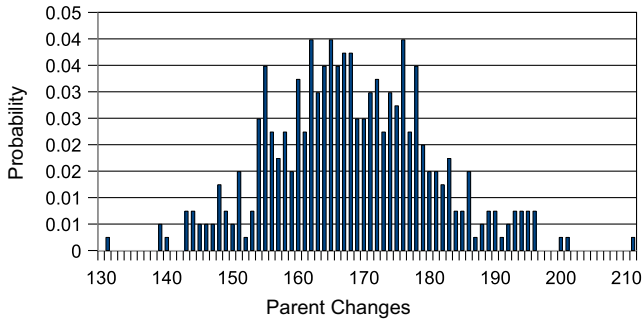
Fig. 12 shows the absolute difference in packet reception ratio between experiment and simulation (PRR error) for several representative experiments. Similarly, Fig. 13 shows the KW distances between the overlapping portions of CPDFs of real signal power simulation and those generated using the suggested simulation model. Both figures show examples of various link qualities encountered in low-power wireless network deployments out of the 21 partitions used in this evaluation. For all simulations tested, at least one of the phase assumptions and two time-scales led to a PRR error of at most 0.11. At least four time-scales led to this bound in all but one of the 21 partitions examined. Among all time-scales tested and all partitions, 81% of the experiments tested led to a PRR error of at most 0.11. The KW distance was below 0.15 for 69% of the links tested. The maximum KW distance for any experimental partition with an optimal phase assumption was 0.23. Such high distances may be the result of a CPDF with few entries that was observed in some of these experiments. However, for many of these experiments the KW



(a) PMF of parent changes using the real signal power model in buckets of 1. The median number is 60.



(b) PMF of parent changes using the constant (mean) signal power model in buckets of 1. The median number is 1.



(c) PMF of parent changes using the CPM signal power model in buckets of 1. The median number is 168.

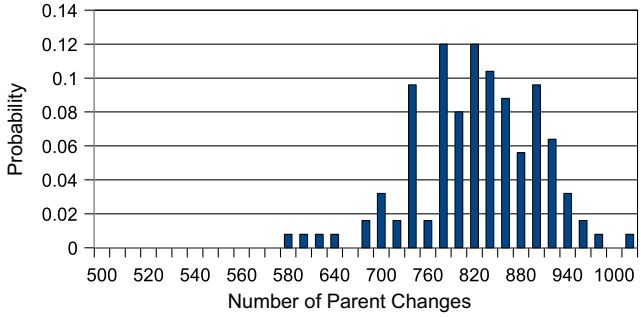
Fig. 16. A probability mass function (PMF) of the number of parent changes using the CPM, constant, and real signal power models for the first simulation topology. Note the differences in scale on the axes of each plot; the effects of burstiness on parent changes are apparent.

distance was as low as 0.01, indicating that burstiness is captured at the time-scale of an individual packet by this simulation model.

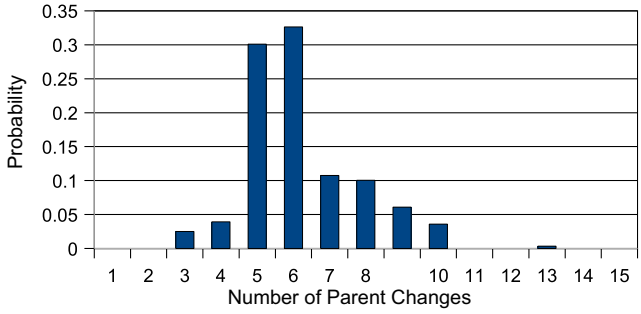
7. Effects on the simulation of higher-level protocols

Low-power wireless networks can benefit substantially from the proposed model in practice due to pervasive burstiness. Unlike models that assume link quality to be independent in time, our model takes time dependence into account and can thus lead to more accurate simulation of the dynamics of an application's behavior. As a result, this simulation model can more correctly capture the behavior of higher-level protocols.

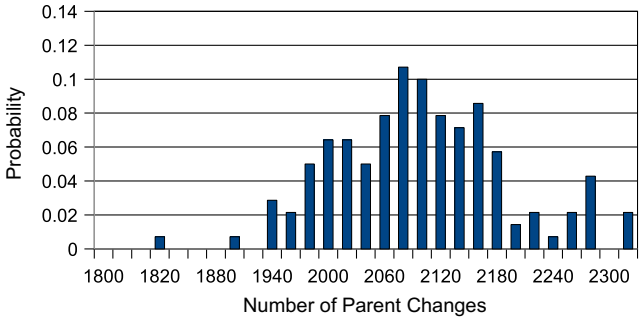
To demonstrate this, we conduct two simulations of distinct four-node topologies. The metric of evaluation is the number of *parent changes* in the standard TinyOS 2.0 collection layer, CTP [25]. CTP builds trees for routing data from source nodes to sink nodes, which act as roots of the collection trees. When a parent change happens, routing destination of a particular node (i.e., the node's parent in the collection tree) changes to a different node. In the results that follow, we consider the initial parent assignment for each source and any subsequent changes to the sources' parents according to the protocol as a parent change. For each simulation, we illustrate a probability mass function over the number of parent changes in CTP using several strategies for modeling signal power.



(a) PMF of parent changes using the real signal power model in buckets of 20. The median number is 816.



(b) PMF of parent changes using the constant (mean) signal power model in buckets of 1. The median number is 6.



(c) PMF of parent changes using the CPM signal power model in buckets of 20. The median number is 2070.

Fig. 17. A probability mass function (PMF) of the number of parent changes using the CPM, constant, and real signal power models for the second simulation topology (Fig. 15). Note the differences in scale on the axes of each plot; the effects of burstiness on parent changes are apparent.

In the first simulation, we consider a topology of one source and three sinks. In this simple scenario, the source is connected to all the sinks using a model of the Gates Building wireless link considered above; this topology is shown in Fig. 14. However, the sinks are not connected and cannot hear each other. We model noise using the CPM algorithm [7] and a local noise trace. For signal power, we take the constant signal power model (the mean signal power), the CPM model at $f = 100$ Hz (a period of 10 ms) with the out of phase assumption and the EVP filling-in algorithm, and the real signal power for comparison. The source sends 100,000 packets as fast as possible achieved by calling `send()` in `sendDone()` in the TinyOS program.

For the second simulation, we also consider a topology of four nodes. In this case, there are three sources and one sink. Each node is connected to all of the others using a simulation of the Gates Building trace discussed in the previous section; this topology is shown in Fig. 15. As before, we model noise using the CPM algorithm and a local noise trace, and we use the same models for signal power described in the first simulation. Over this network, we send 100,000 packets from each source to the sink node as fast as possible, for a total of 300,000 packets. Contrasting the two simulations studied, we see that this second simulation allows packets to route through siblings and also sends more packets while fixing the sink node.

The results of both simulations clearly show that burstiness impacts the behavior of the higher-level protocol. The PMFs for each signal power model are given in Fig. 16 for the first simulation and Fig. 17 for the second simulation. For the first simulation topology (Fig. 14), the median number of parent changes is 60 for the real signal power model. For the constant signal power, the median number of parent changes is only 1 (the initial assignment of the parent), while using CPM signal power, the median number is 168. Thus, the CPM model overestimates the number of parent changes by a factor of about 2.8 while the constant signal power model underestimates the number of parent changes by a factor of 60, considering only the initial parent assignment in 76% of the cases. For the second simulation (Fig. 15), the median number of parent changes for the real signal power is 816 parent changes over the course of the simulation, while for the constant signal power it is 6 (of which three are initial assignments for the three sources) and for CPM signal power it is 2070. While the CPM model overestimates the number of parent changes by a factor of about 2.5, this is a significant improvement over the factor-of-136 underestimation error in the constant signal power model.

These improvements are especially significant when we consider that it is better for a simulator to be pessimistic than optimistic. Optimistic simulators can give protocol developers false confidence and thus lead to protocols that fail to work in practice. On the other hand, pessimistic simulators encourage the development of more robust protocols that work well in practice since conditions are better than those expected in the simulation phase of the protocol's development.

8. Conclusions

This paper presents an improvement to low-power wireless simulation by suggesting a way to model the signal power of wireless links. In particular, we suggest two algorithms to fill in for signal traces that are inherently missing from experiment. We make three assumptions about phase. By examining the PRR of links in simulation and experiment, we note that choosing correct parameters in the suggested model can lead to very low PRR error as compared to the experimental trace. Using the KW distance, we show that our simulation technique preserves the packet reception correlation as compared to a real signal power simulation. In addition, we study the time-scale characteristics of this simulation model. Our results indicate that the model is effective regardless of time-scale at which the initial learning experiment was collected, ensuring a greater level of freedom of users of these models in real simulation environments. Finally, we show that the proposed algorithms lead to a more accurate simulation of higher-level protocols.

The algorithms presented in this paper, along with the CPM algorithm that we suggested in previous work, can be used to create a high-fidelity simulation of low-power wireless links. Such accurate models and simulations can help protocol and application developers learn about the effectiveness of their algorithms and implementations by collecting traces of signal power and noise from the intended deployment environment.

Acknowledgments

We would like to thank Kannan Srinivasan for insightful discussions. Portions of this work were conducted at the Wireless Networks Laboratory, Cornell University, under the direction of Zygmunt Haas. This work was supported by generous gifts from the Bartels Family, Microsoft Research, Cisco Research, Intel Research, DoCoMo Capital, Foundation Capital, the National Science Foundation under Grants #0615308 ("CSR-EHS") and #0627126 ("NeTS-NOSS"), and a Stanford Terman Fellowship.

References

[1] P. Levis, TinyOS: an open platform for wireless sensor networks, invited tutorial, in: Proceedings of the IEEE Seventh International Conference on Mobile Data Management (MDM'06), 2006.
[2] S. Fortune, D. Gay, B. Kernighan, O. Landron, R. Valenzuela, M. Wright, WiSE design of indoor wireless systems: practical computation and optimization, IEEE Computational Science and Engineering 2 (1) (1995) 58–68.
[3] Wireless Valley Software, 2003–2005. <http://www.connect802.com/w-valley.htm>.
[4] Mentum Inc., 2008. <http://www.mentum.com/>.
[5] D.S.J. De Couto, D. Aguayo, J. Bicket, R. Morris, A high-throughput path metric for multi-hop wireless routing, in: Proceedings of the Ninth Annual International Conference on Mobile Computing and Networking (MobiCom'03), 2003, pp. 134–146.
[6] N. Gershenfeld, The Nature of Mathematical Modeling, Cambridge University Press, Cambridge, 1999.
[7] H. Lee, A. Cerpa, P. Levis, Improving wireless simulation through noise modeling, in: Proceedings of the Sixth International Conference on Information Processing in Sensor Networks (IPSN'07), 2007, pp. 21–30.

[8] C. Givens, R. Shortt, A class of wasserstein metrics for probability distributions, Michigan Mathematical Journal 31 (2) (1984) 231–240.
[9] P. Levis, D. Gay, V. Handziski, J. Hauer, B. Greenstein, M. Turon, J. Hui, K. Klues, C. Sharp, R. Szewczyk, et al., T2: A Second Generation OS for Embedded Sensor Networks, Technical Report, TKN-05-007, Telecommunication Networks Group, Technische Universitat Berlin, 2005.
[10] P. Levis, N. Lee, M. Welsh, D. Culler, TOSSIM: accurate and scalable simulation of entire TinyOS applications, in: Proceedings of the First International Conference on Embedded Networked Sensor Systems (SenSys'03), 2003, pp. 126–137.
[11] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, et al., TinyOS: an operating system for sensor networks, in: W. Weber, J. Rabaey, E. Aarts (Eds.), Ambient Intelligence, Springer, 2005.
[12] M. Zuniga, B. Krishnamachari, Analyzing the transitional region in low power wireless links, in: Proceeding of the First IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON'04), 2004, pp. 517–526.
[13] A. Goldsmith, Wireless Communications, Cambridge University Press, Cambridge, 2005.
[14] S. Bae, K. Kim, Analysis of wireless link for mobile sensor network, in: Proceedings of ICEE'06, 2006.
[15] D. Lai, A. Manjeshwar, F. Herrmann, E. Uysal-Biyikoglu, A. Keshavarzian, Measurement and characterization of link quality metrics in energy constrained wireless sensor networks, in: Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM'03), 2003, pp. 446–452.
[16] TinyOS Repository, 2007. (<http://www.tinyos.net/>), `/tinyos-2.x-contrib/stanford-sing/apps/RssiSample`, 2007.
[17] T. Rusak, P. Levis, Poster abstract: on the scaling properties of low power wireless links, in: Proceedings of the Sixth ACM Conference on Embedded Network Sensor Systems (SenSys'08), 2008, pp. 441–442.
[18] T. Rusak, P. Levis, Burstiness and scaling in the structure of low-power wireless links, SIGMOBILE Mobile Computing Communication Review 13 (1) (2009) 60–64.
[19] J. Polastre, R. Szewczyk, D. Culler, Telos: enabling ultra-low power wireless research, in: Proceedings of the Fourth International Symposium on Information Processing in Sensor Networks (IPSN'05), 2005, pp. 364–369.
[20] CC2420 Datasheet, Texas Instruments Inc., 2007. <http://www.focus.ti.com/docs/prod/folders/print/cc2420.html>.
[21] K. Srinivasan, M. Kazandjieva, S. Agarwal, P. Levis, The beta-factor: measuring wireless link burstiness, in: Proceedings of the Sixth ACM Conference on Embedded Network Sensor Systems (SenSys'08), 2008, pp. 29–42.
[22] C. Metcalf, TOSSIM Live: Towards a Testbed in a Thread, Master's Thesis, Colorado School of Mines, 2007.
[23] Y. Rubner, C. Tomasi, L. Guibas, A metric for distributions with applications to image databases, in: Proceedings of the Sixth International Conference on Computer Vision, 1998, pp. 59–66.
[24] D. Aguayo, J. Bicket, S. Biswas, G. Judd, R. Morris, Link-level measurements from an 802.11b mesh network, in: Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM'04), 2004, pp. 121–132.
[25] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, P. Levis, Collection Tree Protocol, Technical Report, SING-09-01, Stanford Information Networks Group, 2009.

Tal Rusak is a graduate student at the Computer Science Department at Stanford University, funded by the NDSEG and Tau Beta Pi Nagel fellowships. Tal's research interests lie in understanding the structure and dynamics of networks and novel computing systems. He received a Bachelor's Degree in Computer Science summa cum laude from Cornell University in 2009. Tal was recognized as the 2009 Computing Research Association (CRA) Outstanding Undergraduate Award Winner and received the 2009 Cornell Computer Science Prize for Academic Excellence. His work was also recognized by the Best Paper Award at ACM MSWiM'08 and two first prizes at ACM Student Research Competitions.

Philip Levis is an Assistant Professor of Computer Science and Electrical Engineering at Stanford University. He researches wireless networks, low power computing, sensor networks, and networked systems infrastructure. He received his Sc.B. in 1999 from Brown University, his M.S. in 2001 from the University of Colorado at Boulder, and his Ph.D. from UC Berkeley in 2005. In 2008 he was named one of five Microsoft New Faculty Fellows and also received an NSF CAREER grant.