A Simple Mechanism for Capturing and Replaying Wireless Channels

Glenn Judd and Peter Steenkiste
Carnegie Mellon University
Pittsburgh, PA, USA
glennj@cs.cmu.edu prs@cs.cmu.edu

ABSTRACT

Physical layer wireless network emulation has the potential to be a powerful experimental tool. An important challenge in physical emulation, and traditional simulation, is to accurately model the wireless channel. In this paper we examine the possibility of using on-card signal strength measurements to capture wireless channel traces. A key advantage of this approach is the simplicity and ubiquity with which these measurements can be obtained since virtually all wireless devices provide the required metrics. We show that for low delay spread environments wireless traces gathered using this method can be replayed in a physical wireless emulator to produce higher layer network behavior that is similar to the behavior that would have occurred in the real world. Thus, wireless channel traces gathered using on-card metrics are an effective means of enabling existing low delay spread wireless testbeds to be emulated.

1. INTRODUCTION

Despite concerns regarding their shortcomings in terms of realism, wireless simulators have remained popular due to the control, repeatability, and ease-of-use that they afford researchers. Recent experiences with wireless testbeds, however, have confirmed that these experimental benefits come with the cost of inaccurate results.

We are developing a physical layer wireless emulator [1] that gives us complete control over the physical wireless channel. Like wireless simulators, our emulator allows us to run experiments in a controlled and repeatable virtual wireless environment. Like wireless testbeds, however, this approach also allows us to to run real applications on real wireless hardware.

With the power of complete wireless channel control comes the challenge of accurately modeling channel behavior. One possible approach is to utilize statistical models of wireless channel behavior. Clearly this approach is simple; it may, however, yield less realism than we desire. A more sophisticated approach is to use wireless channel sounding equipment to precisely characterize a real wireless channel. This is ideal from a realism standpoint, but the high expense of such equipment prevents its widespread use.

SIGCOMM'05 Workshops, August 22–26, 2005, Philadelphia, PA, USA.

In this paper we examine the possibility of using on-card signal strength measurements to capture wireless channel traces. A key advantage of this approach is that every wireless device that provides on-card signal strength statistics (virtually all do) can be used to measure wireless channels in situ. We show that in low delay spread environments wireless traces gathered using this method can be replayed in our wireless emulator to produce higher layer behavior that is similar to the behavior that would have occurred in the real world.

While we limit our discussion to the efficacy of this channel modeling technique in a physical emulator, this same technique could also be applied in traditional wireless simulators.

Section 2 provides a brief description of our physical wireless emulator. Section 3 discusses how on-card measurements of signal strength can be used to gather and replay wireless channel traces, and Section 4 then compares higher layer performance using this channel replay technique to real-world higher layer performance. Section 5 discusses limitations of our approach as well as potential enhancements. Related work is presented in Section 6 followed by our conclusion in Section 7.

2. EMULATOR OVERVIEW

We now briefly describe the architecture of our emulator and our current implementation of that architecture. For a more detailed discussion see [1].

2.1 Architecture

The architecture of our emulator is shown in Figure 1. A number of "RF nodes" (e.g. laptops, access points, cordless phones, or *any* wireless device in the supported frequency range) are connected to the emulator through a cable attached to the antenna port of their wireless line cards. For each RF node, the RF signal transmitted by its line card is "mixed" with the local oscillator (LO) signal. This shifts the signal down to a lower frequency where it is then digitized, and fed into a DSP Engine that is built around one or more FPGAs. The DSP Engine models the effects of signal propagation (e.g. large-scale attenuation and small-scale fading) on each signal path between each RF node as depicted in Figure 2. Finally, for each RF node, the DSP combines the appropriately processed input signals from all the other RF nodes. This signal is then sent out to the wireless line card through the antenna port.

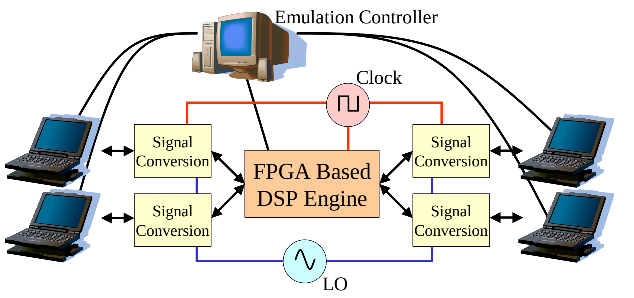

Figure 1: Emulator Architecture

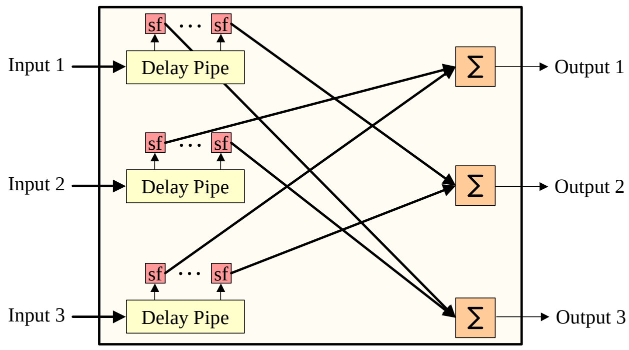

Figure 2: Typical DSP Engine Operation

The operation of the emulator is managed by the Emulation Controller which coordinates the movement of RF nodes (and possibly physical objects) in the emulated physical space. The Emulation Controller uses location information (and other factors as dictated by the signal propagation model in use) to control the emulation of signal propagation within this emulated environment. In addition, the Emulation Controller coordinates node (and object) movement in physical space with the operation of RF node applications and sending of data.

2.2 Implementation

A proof-of-concept prototype of this architecture was presented in [2]. We are in the process of implementing a much improved "Version 2" implementation of this architecture.

Our Version 2 DSP Engine is currently under development. The Version 2 Signal Conversion Module, however, is complete and functional. The A/D and D/A boards used in this module are capable of running at 210 Msps. This allows us to capture around 100 MHz of bandwidth directly, and is sufficient to capture all North American 802.11b/g channels or a portion of 802.11a.

The Version 2 Signal Conversion Module utilizes a modest FPGA, which allows each module to assist the DSP Engine in certain cases. This FPGA allows us to use two Signal Conversion Modules to validate our production emulator by emulating two RF channels. In this case, each module implements a single RF channel. While the scale of this approach is limited compared to a version that uses a real DSP Engine, the fidelity of channel emulation is the same. Hence, the results we obtain here will apply to our completed emulator.

3. TRACE CAPTURE AND PLAYBACK

3.1 Trace Capture

Figure 3 shows our approach for gathering traces of signal strength. A transmitter constantly sends very small 802.11 broadcasts using a low modulation rate (we use 2 Mbps). As deep signal fades may prevent sounding packets from being received successfully, packets are tagged with sequence numbers that enable us to detect when packets fail to be received. The receiver operates in "monitor mode". This mode gives the receiver complete 802.11 layer packet information. The receiver logs all captured packets from the transmitter including measurements of received signal strength (RSSI) and noise. This trace is then post-processed to generate a file that lists time using the MAC timestamp and received signal strength. This post-processing replaces missing packets - inferred from missing sequence numbers - with low RSSI values (we currently use an RSSI of -1). Using this approach we are able to record RSSI samples with a granularity of approximately 2 ms.

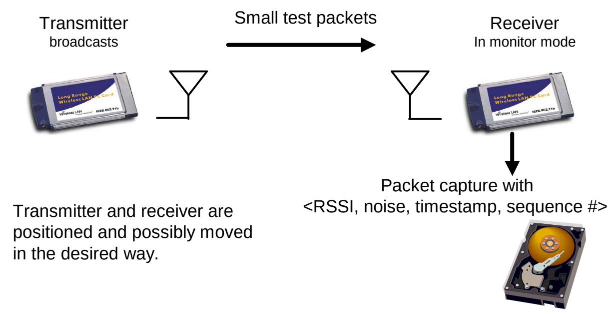

Figure 3: RSSI-based Channel Capture

For our experiments, we utilize Engenius NL-2511 Plus EXT2 cards based on the Prism2.5 chipset as well as an Atheros 5212 based card. The Atheros card was used for RSSI measurement only. These cards measure received signal strength at the beginning of packet acquisition, so our RSSI samples are quick samples rather than an average of RSS for the whole packet.

Figure 4 plots a sample of a signal strength trace. This particular trace was captured with the receiver antenna mounted on a car parked at the side of a freeway while the transmitter drove by at approximately 60 MPH. From this trace we see that the transmitter and receiver had a good line-of-sight connection when the cars were at their closest point. At further distances signal strength degrades and fading increases.

3.2 Trace Playback

Once we have obtained a trace of signal strength, we can replay this trace in our emulator. To do this, the Emulation Controller reads the trace and replays it in real time. That is, for each $< RSS, timestamp >$ pair in the sample, the Emulation Controller waits until the emulation time matches the recorded time and then commands the Emulator to set the emulated path loss to match the observed path loss. The temporal resolution of the channel power settings is limited by the trace recording process which is 2 ms as discussed above.

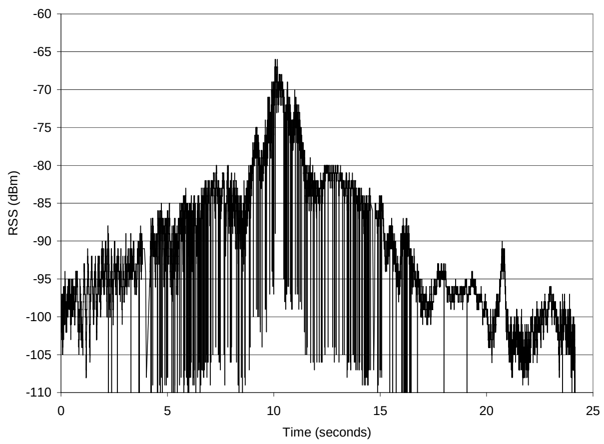

Figure 4: Sample Channel Trace

3.3 Limitations

Our approach is attractive in that it is supported by commodity hardware currently found in wireless testbeds. Using commodity hardware, however, has limitations that we can work around to some degree such as: non-linearities in RSSI measurements; bogus RSSI values; missing RSSI values in deep fades; and a lack of foreign RF interference characterization; In addition there are fundamental limitations to our approach with current commodity hardware such as: lack of channel impulse response/multipath information; path loss limited by accuracy of RSSI measurement and transmit power consistency; and sounding temporal resolution limited by packet transmit rate. We will discuss several of these issues further in Section 5.

4. COMPARISON WITH REAL-WORLD BEHAVIOR

4.1 Methodology

Our method is clearly straightforward and can easily be used to gather traces from many existing wireless testbeds. An important question is how much realism we lose with respect to the real world. Clearly our technique does not completely compute the impulse response of the channel and track it over time. This would require a full-blown channel sounder. We do, however, track the RSS changes due to large scale path loss and small scale fading with 2 ms granularity. We now show that in low delay spread environment, these metrics are sufficient to produce link-level behavior that is quite similar to real-world behavior.

To show this, we conducted an experiment designed to allow us to simultaneously measure real-world link-layer performance while gathering a signal strength trace. The idea is that we can then replay the captured signal trace while re-running the link-layer test. We hope to observe similar performance. Note that this is an ambitious goal since even if we could perfectly reproduce the radio channel that existed when the original link-layer test was conducted, factors outside of our control will lead to inevitable variance from the original test during a replay.

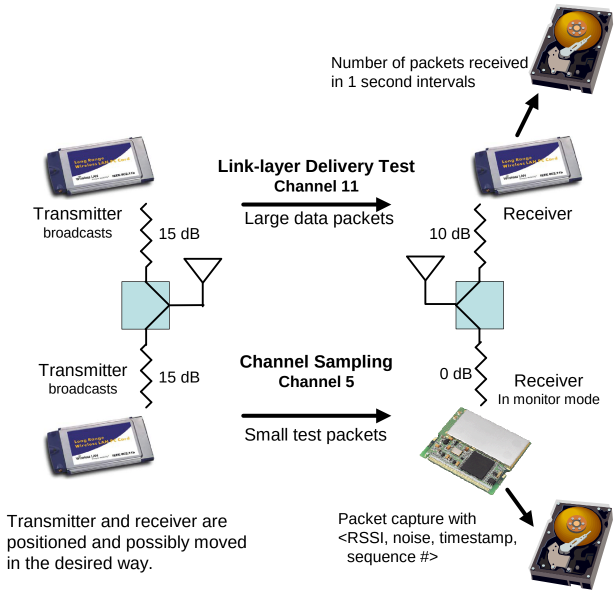

Figure 5: Link-layer Test/Channel Capture

Figure 5 shows our setup. In this experiment, we run two concurrent tests: a link-level behavior test, and a channel measurement test. Each test uses a distinct transmitter, receiver pair. To ensure that the channel is as similar as possible, we connect both transmitters to the same antenna via a splitter/combiner. Each transmitter operates on a non-overlapping 802.11 channel; this allows us to conduct the link-level and channel measurement experiments concurrently. We introduce some amount of attenuation to further avoid interference between transmitters. The receivers are setup in a similar fashion though they require less attenuation since they will only be receiving traffic. Note that we use less attenuation on the channel measurement receiver. This allows us to measure the channel even when no packets can be received at the receiver. In addition, the channel transmitter uses more power than the test transmitter to further increase our ability to measure the channel when the test receiver cannot receive packets.

4.2 Two-channel Measurements

While using different channels allows us to simultaneously run applications and gather signal strength traces, there is still likely to be some divergence between the two channels. We stress that this divergence does not affect our proposed trace gathering and replaying approach in any manner. Rather, it only pessimistically affects our ability to verify the accuracy of our approach.

We now explore what we lose in gathering two signal strength traces simultaneously. To do this, we replace the delivery test in Figure 5 with another signal strength capture. Hence, for the following tests, we are simply running two channel measurements concurrently.

We first compared cross channel performance when using a coaxial and variable attenuator setup in place of the over the air setup shown in Figure 5. We found that as expected, cross channel RSSI measurements are nearly identical in this case.

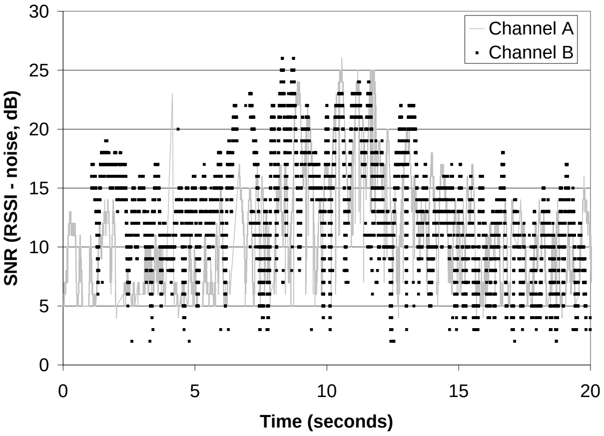

Figure 6: Two-channel Capture - Over-the-air

We then repeated this test over the air. The traces from this test are shown in Figure 6. In this case, the traces are not identical for three reasons: 1 - the transmitters are not synchronized so the channel is being sampled at different times, 2 - some frequency selective fading is occurring, 3 - RSSI measurement error. Nevertheless, the traces are similar enough for our purposes though they will introduce some divergence between our emulated results and real-world results. Hence, our comparison of real-world link-layer performance and the emulated replay will be slightly pessimistic since a single channel capture will not have this variation.

4.3 Comparison Results

We now compare the performance of real-world link-layer behavior vs. an emulated playback of this same behavior. The link layer test that we conducted for this comparison was to send approximately 124 large (1460 bytes payload) UDP broadcast packets per second from the test transmitter to the test receiver. As previously discussed, we concurrently measured the wireless channel as shown in Figure 5. We were able to obtain approximately 2-3 channel samples per test packet. We then replayed this test in the emulator for comparison.

Figure 7 and Figure 8 show the results of two separate record/ playback verification tests. With a few notable exceptions, the results are quite similar. The average packets received in the emulated replays generally closely tracks the original results. This is in spite of extraneous error introduced by our two-channel verification technique, imperfections in card characterization, variation in packet send time, and other similar factors.

For each 1 second interval in these tests, we computed the absolute difference between the real-world throughput and the throughput in the corresponding emulated interval. The CDF of these error measurements is shown as the "Atheros" plot in Figure 9. This figure also shows the CDF of three tests (not shown above) comparing real-world throughput vs. emulated throughput where we used a Prism 2.5 card for channel sounding instead of the Atheros card used above. In both cases we see that the majority of time intervals were reproduced with low error. There are, however, some time intervals with significant error. As a result, it is possible to construct movement patterns where our verification tests will yield poor results.

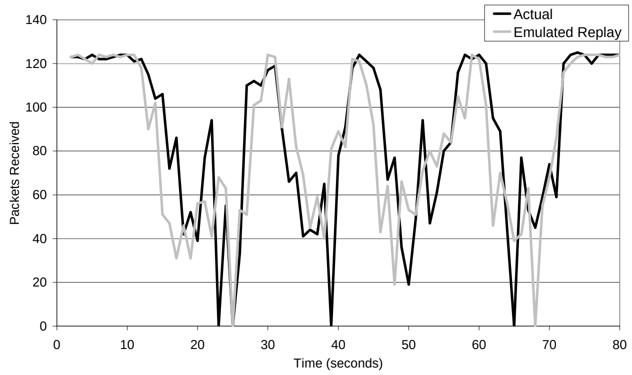

Figure 7: Real-world vs. Emulated Replay

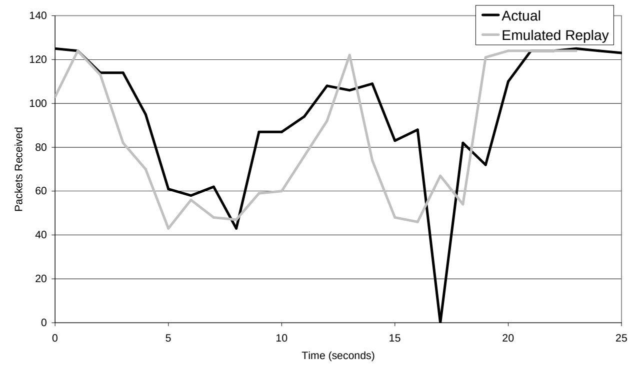

Figure 8: Real-world vs. Emulated Replay

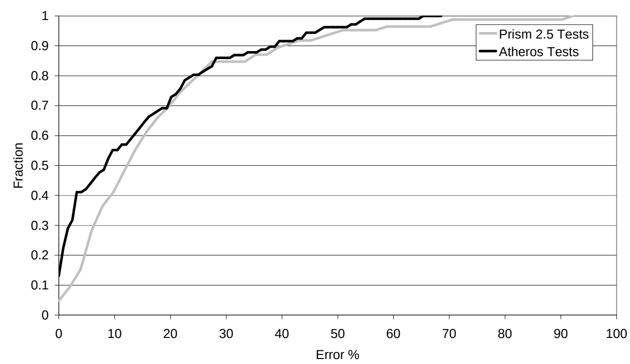

Figure 9: CDF of Test Error

5. DISCUSSION

5.1 RSSI Considerations

In order to effectively translate RSSI measurements into path loss measurements, we must process the received RSSI measurements to remove imperfections in the measurements. We now discuss two significant imperfections that must be accounted for.

RSSI Non-linearity. As discussed in [1] on-card received signal strength measurements (RSSI) are not completely accurate even under the best circumstances. Thus, relying strictly on RSSI for trace playback without a mapping between RSSI and RSS (the actual received signal strength) will distort the replayed signal. The effect of this can be reduced by characterizing the RSSI-RSS relationship on a per-card basis. Figure 10 shows the mean RSSI measured by an Atheros-based card as actual RSS is varied using our emulator. Ideally each card in a testbed would be characterized in this manner. At the very least, each type of card should have an RSSI characterization performed.

An important feature of Figure 10 that should be taken into consideration is the fact that RSSI values near the lowest end of the card's reception range become indistinquishable. As a result, channel characterization will be less accurate in this regime.

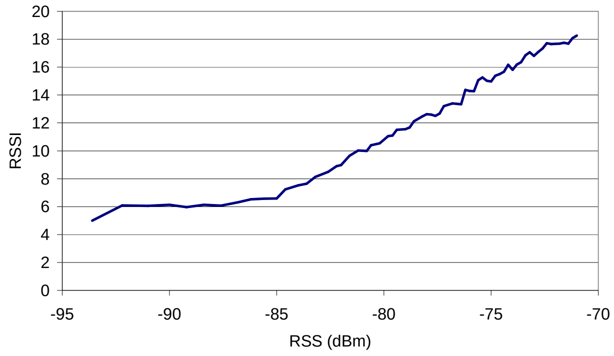

Figure 10: Card Characterization

"Bogus" RSSI Values. In addition to the non-linearities shown in Figure 10, wireless cards tend to return a certain number of RSSI values that seem to be bogus. Our experience has been that both Atheros and Prism 2.5 cards occasionally return values that are around 20 dB below what seems to be the true value.

In order to get a good match between our real-world comparisons discussed earlier, we found it necessary to filter out these bogus values. We did this by limiting the rate at which the signal strength was allowed to change, and interpolating between "good" values. Figure 11 shows the raw signal trace used in the Figure 8 test versus its "corrected" counterpart.

The benefit of this RSSI correction is shown by comparing the playback result of the uncorrected raw version. Figure 12 shows the same test as that in Figure 7, but with the use of raw RSSI values. Clearly eliminating these bogus RSSI values has yields a significant improvement in matching the real-world measurements.

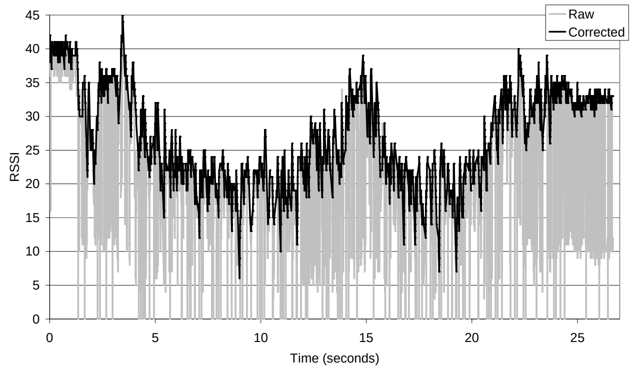

Figure 11: RSSI Correction

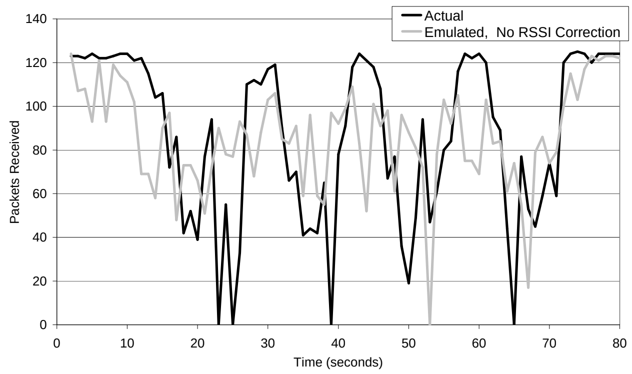

Figure 12: Raw RSSI Emulation Accuracy

5.2 Noise

Conspicuously missing from our trace playback methodology is use of the noise values reported by the card. These noise values are the sum of both true noise as well as interference. True noise comes from both sources external to the card - most importantly thermal radiation - as well is sources internal to the card which make up the card's "noise figure". Interference that comes from sources internal to the network is simply non-captured traffic; we call this "internal interference". We call "external interference" received signals from RF sources that are outside of experimental control.

Externally generated true noise is likely to be mostly due to thermal radiation and constant. Hence, this can usually be computed rather than measured. As the card's internal noise figure and uncaptured traffic are naturally occurring features of playback recording them is not useful. It might be useful, however, to know levels of external interference. This value would, however, need to be separated out from the noise figure and internal interference. In many cases, the difficulty of this task is not worth the added fidelity that it would provide. For networks with significant external interference, however, the on-card noise measurements could potentially provide a means of emulating this interference.

5.3 Improving Channel Recording and Playback

We now discuss a few additional sources of error in our current trace recording and playback methodology and how these might be addressed.

Channel Probe Granularity. We currently use simple UDP broadcast packets to probe the channel. Our granularity is limited to 2 ms using this approach. By using 802.11 level packets, with a short preamble we should be able to increase our resolution. In addition, some NICs allow the 802.11 CSMA/CA mechanism to be turned off. This could be used to greatly decrease inter-packet delay and greatly increase sampling resolution.

Multipath. Finer granularity measurements will improve our ability to capture fast fading induced by multipath effects. Our technique is not, however, amenable to analyzing radio-level effects such as the efficacy of a RAKE receiver or equalizer. This level of channel modeling fundamentally requires a channel sounder that can capture the impulse response of the channel.

A related question is why multipath effects do not make our technique ineffective considering measurements [6] that show multipath can dominate RSS in certain situations. In our case, the delay spread of our network is well within the radio's capabilities. As shown in [6] multipath does not affect packet reception very much for low delay spreads. As a result, our technique should work well for environments that are within a radio's ability to mitigate multipath effects. Outside of that regime, however, our technique will be less effective. Additional work is required to quantify our technique's accuracy in higher delay spread environments.

5.4 Network Modeling

Our experiments have demonstrated channel capture and playback of a single channel. This technique can be extended to an entire wireless network in several ways. First, if the channels are relatively stable or correlation between channels is low, each channel in the network can simply be captured independently in time. If channel correlation needs to be captured, these measurements must occur concurrently. In this case, transmitters that are nearby must take turns in sending probe packets; for 802.11 networks this can largely be accomplished simply by using 802.11's CSMA/CA mechanism. In some cases, it may be necessary to control the rate of probe packets in order to reduce the likelihood of collision of probe packets at distant receivers.

Once traces have been obtained for all channels in the network, playback proceeds in the same manner as before.

5.5 Multi-element Air Interfaces

The technique that we have presented relies on on-card signal metrics for channel characterization. In order for this technique to be useful on future multi-element NICs, these NICs must provide per-element channel information. If hardware vendors provide access to this information, our technique should apply to these emerging devices.

6. RELATED WORK

Commercially available channel sounders [3] provide a powerful means of recording rich channel information. Commercial channel emulators [4, 5] can be used to replay a small number of channels with excellent fidelity. Unfortunately, the cost of these devices is prohibitive. Moreover, the widespread deployment of these devices into existing testbeds is impractical. We have shown that the simple technique of using on-card channel measurements can produce results that are sufficiently accurate to produce realistic link-layer performance.

Numerous researchers (e.g. [6]) have used on-card channel measurements to analyze channel behavior. [7] uses on-card channel measurements to drive a simulator for the purpose of network troubleshooting; these measurements are at a fairly coarse granularity since they are not targeting the level of realism required by physical emulation. We are not aware of any previous efforts that have used on-card measurements for the purposes of physical emulation, or to demonstrate that on-card measurements can provide realistic higher layer performance.

7. CONCLUSION

Accurate wireless channel modeling is an important element in physical layer wireless emulation as well as wireless simulation. We have presented a simple method for gathering traces of wireless channel behavior. We further developed a technique of analyzing the effectiveness of our channel emulation by simultaneously recording a signal strength trace while a real application is being run on the same transmit and receive antennas. Using this technique, we have shown that the wireless signal traces we gather can produce behavior that is surprisingly similar to real-world wireless behavior in spite of the simple nature of on-card channel measurements.

8. REFERENCES

[1] G. Judd and P. Steenkiste. Using Emulation to Understand and Improve Wireless Networks and Applications. *Proc. of NSDI 2005. Boston, MA*, May 2005.

[2] G. Judd and P. Steenkiste. Repeatable and Realistic Wireless Experimentation through Physical Emulation. *Proc. of HotNets-II. Cambridge, MA*, November 2003.

[3] PROPSound. Propsound multi-dimensional channel sounder, `http://www.elektrobit.ch/products/propsound/`.

[4] PROPSim. Propsim c8 wideband multichannel simulator, `http://www.propsim.net/`.

[5] Spirent Communications. Tas4500 flex5 rf channel emulator, `http://www.spirent-communications.com/`.

[6] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris. Link-level Measurements from an 802.11b Mesh Network. *Proc. of SIGCOMM 2004. Portland, OR*, August 2004.

[7] L. Qiu, P. Bahl, A. Rao, and L. Zhou. Troubleshooting multihop wireless networks. Tech. Rep. MSR-TR-2004-11, Microsoft Corporation, November 2004.