T-SIMn: Towards the High Fidelity Trace-Based Simulation of 802.11n Networks

Ali Abedi
University of Waterloo
ali.abedi@uwaterloo.ca

Andrew Heard
University of Waterloo
asheard@uwaterloo.ca

Tim Brecht
University of Waterloo
brecht@cs.uwaterloo.ca

ABSTRACT

In this paper, we describe the design, implementation and evaluation of a new framework for the trace-based evaluation of 802.11n networks, which we call T-SIMn. We first develop novel techniques for collecting and processing traces for 802.11n networks that incorporate Frame Aggregation (FA). We then demonstrate that the simulator portion of our framework (SIMn) accurately simulates throughput for one, two and three-antenna Physical Layer Data Rates (PLDRs) in 802.11n with FA. Finally, we evaluate the T-SIMn framework (including trace collection) by collecting traces using an iPhone which is representative of a wide variety of one antenna devices. We show that our framework can be used to accurately simulate these scenarios and we demonstrate the fidelity of SIMn by uncovering problems with our initial evaluation methodology. We expect that the T-SIMn framework will be suitable for easily and fairly comparing algorithms that must be optimized for different and varying 802.11n channel conditions which are challenging to evaluate experimentally. These include rate adaptation, frame aggregation and channel bandwidth adaptation algorithms.

1. INTRODUCTION

With billions of WiFi devices now in use, combined with the rising popularity of high-bandwidth applications such as streaming video, demands on WiFi networks continue to rise. The 802.11n standard introduced several new physical layer features including MIMO, 40 MHz channels, denser modulations, and a shorter guard interval, to increase throughput. We refer to the combination of features as a *rate configuration*. Combinations of these features results in up to 128 different rate configurations. In order to optimize throughput in an 802.11n network, we must choose the rate configuration that results in the best trade-off between Physical Layer Data Rates (PLDRs) and error rates, which is highly dependent on environmental conditions. Because the radio spectrums are shared by WiFi devices included in computers, cell phones and tablets; as well as Bluetooth devices, wireless keyboards/mice, cordless phones, and microwave ovens, it is challenging to experimentally evaluate and compare the performance of WiFi networks. Therefore, we need new techniques for understanding and evaluating how to best use 802.11n features.

Previously, we proposed a solution [2] that uses traces to capture environmental conditions, rather than models, to simulate 802.11g networks with high fidelity. We expected that extending this solution to 802.11n networks would be relatively simple. However, it turned out that this is actually a very interesting and challenging problem, due to *MAC layer frame aggregation (FA)* in 802.11n. The complications introduced by frame aggregation required a complete redesign of major components of our trace-based simulator. While the faster PLDRs are an important factor in the throughput gains afforded by 802.11n, it is only when they are used in combination with frame aggregation that 802.11n networks achieve significant increases in throughput when compared with 802.11g. Frame Aggregation (FA) allows multiple MAC frames to be combined into a large physical frame so that they can be transmitted and acknowledged as one aggregated packet, which results in the channel being used more efficiently.

To demonstrate the importance of FA, Figure 1 shows the maximum theoretical throughput obtained using the highest PLDRs in 802.11n for one, two and three spatial streams, respectively. Without frame aggregation throughput is limited to about 50 Mbps. However, when aggregating 32 frames, throughput increases to 350 Mbps.

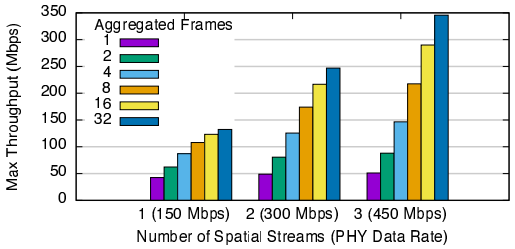

Figure 1: FA: Maximum Theoretical Throughput

Because performance is so heavily dependent on FA, accurate simulation of FA is crucial for T-SIMn to be useful in the study of a range of active research topics. This includes the evaluation of: link adaptation algorithms [15] (which studies physical layer configurations such as rate adaptation [8, 23] and channel bandwidth adaptation [7, 11]) and frame aggregation algorithms [6, 22]. We refer to link adaptation and frame aggregation algorithms collectively as *optimization algorithms*. The contributions of this paper are:

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

MSWiM '16, November 13-17, 2016, Malta, Malta

© 2016 ACM. ISBN 978-1-4503-4502-6/16/11...$15.00

DOI: http://dx.doi.org/10.1145/2988287.2989160

- We design a trace-based simulator (T-SIMn) for the realistic and repeatable performance evaluation of 802.11n networks. We evaluate a prototype implementation of T-SIMn using an iPhone and show that T-SIMn produces highly accurate results.
- Using our framework, we demonstrate that MPDUs within an A-MPDU can experience different error rates which can significantly affect throughput due to the impact losses have on the advancement of the Block-ACK window. We devise a technique for determining and accurately simulating subframe error rates.
- We demonstrate that it is possible to determine the probability of subframe losses within aggregated frames of any length using traces collected with the maximum number of aggregated frames permitted by the rate configuration. We believe this will make it possible to study and fairly compare different FAAs using our framework.

2. BACKGROUND AND RELATED WORK

2.1 Rate Configurations

In 802.11g, a transmission rate can be uniquely identified by the index of the Modulation and Coding Scheme (MCS). This index alone is no longer sufficient to uniquely identify an 802.11n rate, as the transmission rate is now also dependent on the number of Spatial Streams (SSs), the Guard Interval (GI) and the Channel Bandwidth (CB). We refer to this set of parameters as a rate configuration. In the interest of brevity, Figure 2 introduces notation used for describing a rate configuration (sometimes simply referred to as a rate).

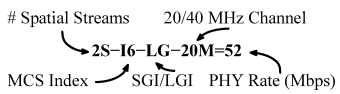

Figure 2: Rate notation

2.2 802.11n Frame Aggregation

Frame aggregation (FA) is a new MAC layer feature in 802.11n that allows multiple frames to be combined to form a larger frame. The 802.11n standard defines two types of frame aggregation: *Aggregated MAC Protocol DATA Unit (A-MPDU)* and *Aggregated MAC Service DATA Unit (A-MPDU)*. These two types of frame aggregation differ by where in the protocol stack aggregation is done. In this paper, we use A-MPDU frame aggregation as it is more widely supported by WiFi devices, including the Atheros devices used in this paper.

By sending multiple MPDUs[1] as an A-MPDU, the *sender* only performs channel sensing, backoff, Distributed Inter-Frame Space (DIFS), Short Inter-Frame Space (SIFS) and wait for an ACK once for the entire set of MPDUs. This results in a greater proportion of time being spent transmitting data, increasing the air time efficiency. The receiver sends back a *Block ACK* which acknowledges all MPDUs at once. In this work, we use MPDU and subframe interchangeably.

2.3 Related Work

Conducting experiments is a common technique for evaluating the performance of WiFi networks. The advantage of this approach is that real-world wireless channel conditions are captured. However, it is challenging to conduct repeatable experiments [3] because channel conditions can vary significantly between trials due to many factors, including mobility, changes in the environment (including the movement of people who are nearby), and the presence of WiFi and non-WiFi interferers [5].

Simulation is a common technique for evaluating the performance of 802.11 networks. Since both the physical and MAC layers are simulated in this approach, comparisons are repeatable. Popular 802.11 network simulators include *ns-3* [16], OPNET Modeler [21] (renamed, Riverbed Modeler) and QualNet [18]. Unfortunately, these simulators may not reflect real-world performance due to the challenging nature of simulating wireless signals in the physical environment. WiFi signals are impacted by many factors, including the distance between devices, material types of surrounding objects and walls, wavelength, mobility and many more [20]. These simulators utilize models for signal propagation, error rates and interference and each model must trade-off computational complexity and accuracy [20]. Additionally, the complexity and time varying nature of all of the factors that can affect a frame's fate makes it incredibly challenging to obtain accurate results, especially since models of one environment (e.g., one home) may not necessarily work in another environment (e.g., an office or even a different home). In the case of environments with mobility, this model may change over time. As a result existing simulators like ns-3 suffer from practical limitations (e.g., ns-3 does not support MIMO). Rather than simulating the physical layer, we collect traces to capture the impact of the physical layer on frame fates, allowing us to handle more complex scenarios than can be accurately modeled by traditional simulators.

Emulation is another alternative for evaluating and studying 802.11 networks. The most common approach is to use real WiFi devices connected by wire to a Field-Programmable Gate Array (FPGA) which simulates signal propagation [12]. An emulation testbed by Judd and Steenkiste [13] is one of the most prominent examples of this type of system. As real devices are used, the MAC and parts of the physical layer are *not* simulated. However, signal propagation *is simulated* using the FPGA to alter the signals being transmitted between devices. The major disadvantage of this approach is that realism is again limited by the models used to simulate the physical layer.

To increase the realism of emulators, hybrid approaches using traces to simulate the physical layer and emulation for the MAC layer have also been proposed [13, 24], however these have been limited to 802.11b networks and rely on measurements of Signal-to-Noise Ratio (SNR) to simulate the physical layer. However, the SNR can not be used to accurately predict frame fates [9]. Instead, we rely on traces to directly capture the impact of the physical layer on frame fates and simulate the well-defined MAC layer. This provides an excellent combination of repeatability and fidelity.

T-RATE [2] is our trace-driven framework for evaluating Rate Adaptation Algorithms (RAAs) designed for 802.11g networks. T-RATE eschews the modeling and emulation of wireless channel conditions in favor of traces that capture channel access and channel error rate information. These traces are used to simulate RAAs using channel conditions limited only by the environment in which traces are collected. Despite the high fidelity of T-RATE, it is limited to the evaluation of RAAs in 802.11g networks. In this paper, we design and evaluate T-SIMn, a more general framework for the trace-based evaluation of 802.11n networks. The

[1]MAC header + MAC payload (IP packet)

most prominent and challenging contribution in T-SIMn is the accurate handling of frame aggregation.

3. THE T-SIMN FRAMEWORK

The main goal of T-SIMn is to achieve repeatability and realism when evaluating the performance of 802.11n networks. To achieve this goal, T-SIMn records all channel conditions that affect throughput in a trace and then uses this trace to simulate different 802.11n optimization algorithms such as link adaptation and frame aggregation. As a result, T-SIM can be used to achieve repeatability by using an identical trace to evaluate different algorithms. In addition, it achieves realism since T-SIMn *relies on traces that are subject to and include information related to actual channel conditions rather than using wireless channel models*, which are known to lack realism [14, 17].

To simulate 802.11n networks with high fidelity, we need to accurately compute the transmission time of a frame and consider all factors that can affect throughput. Computing the transmission time for a frame is a relatively easy task and is done very accurately in our simulator by using timing information available in the 802.11n standard (Section 5.2 provides more details). Environmental factors may affect 802.11 channel access (i.e., CSMA/CA) and channel error rate. If a WiFi or non-WiFi device operating at the same frequency is active during channel sensing, it forces a sender device to back off and therefore limits the number of frames that can be sent. If WiFi or non-WiFi devices interfere with the receiver device, the channel error rate may increase. In T-SIMn, to accurately simulate the time required for frame transmission we need to determine: the delay (overhead) imposed by channel sensing (i.e., CSMA/CA), how long it takes to transmit a frame (including ACK reception and DCF mandatory wait times), and whether or not the transmitted frame is received correctly.

T-SIMn uses two phases to simulate 802.11n. The first phase is trace collection, where a log containing the data necessary for accurately simulating an 802.11n experiment is collected. The second phase is simulation, where the trace is used to determine frame fates, transmission delays and throughput. We now explain these two phases.

3.1 Trace Collection

The purpose of trace collection is to capture channel access and channel error rate information (i.e., a trace), for each 802.11n rate configuration. To enable the simulation of any link adaptation and frame aggregation algorithm at time t, T-SIMn must be able to simulate the transmission of an A-MPDU of any length sent with any chosen rate configuration. This requirement makes trace collection challenging, since at time t, only one particular rate configuration with a specific A-MPDU length can be transmitted. Therefore, no information is available at that time concerning the other combinations of rate configuration and frame lengths.

To address this problem we have designed a trace collection technique that samples all rate configurations in a round robin fashion. After each transmitted frame, the *sender* switches to the next rate in the set; wrapping back around after all rates have been sampled. This round-robin cycling continues for the duration of the trace collection. If we sample all rate configurations within the time a channel is considered stationary. (i.e., the channel coherence time), all configurations experience the same channel conditions. During simulation, to obtain the error rate of a rate configuration at time t, we compute an average error rate for that configuration over a time window centered at t.

If the error rates of the MPDUs within an aggregated frame are identical, we could disable frame aggregation during trace collection and simulate aggregated frames of any size, by simply applying the error rate of (non-aggregated) collected frames to all MPDUs in an A-MPDU. However, a recent study [6] shows that the frame error rate of the MPDUs within an A-MPDU are *not identical*. More specifically, they show that the frame error rate of early MPDUs (i.e., closer to the physical header) are lower than MPDUs that appear latter in the A-MPDU. Although we confirm that individual MPDUs have different error rates, we find patterns other than the increasing error rate pattern (observed in [6]). We discuss these findings in greater detail in Section 5.4.1. Note that this problem was not an issue in our 802.11g framework since 802.11g does not support frame aggregation. One might imagine that a solution would be, during trace collection, to attempt to sample all combinations of A-MPDUs from length 1 (i.e., no frame aggregation) to the maximum number of MPDUs allowed in an A-MPDU. Since this procedure would be required for all rate configurations, it becomes impossible to sample all combinations in a sufficiently short period of time (i.e., within the channel coherence window). As a result, we design a methodology that infers the fate of MPDUs in an A-MPDU from a longer A-MPDU. Therefore, during trace collection, we only need to sample the longest possible A-MPDU for each rate configuration and infer the FERs of any shorter A-MPDU for that rate from the longer A-MPDU.

In order to record the delay imposed by channel access (CSMA/CA), we use Cycle Counter Information (CCI) reported by a modified Ath9k driver as explained in detail in Section 5.3. These counters help us infer the delay caused by WiFi and non-WiFi devices when performing channel sensing. We modify the Ath9k driver to print a log entry after each aggregated frame (A-MPDU) transmission has completed, which includes the fate of each MPDU and the CCI.

3.2 Simulation

The trace processing phase uses a trace to simulate different optimization algorithms. The simulator component of our framework (called SIMn) performs a time based simulation using a trace, where the *sender* saturates the link to the *receiver* by sending as many aggregated frames as possible.

Figures 3 illustrates the work flow of SIMn. Solid lines represent the execution flow of the simulator, with the direction being indicated by a closed (i.e., filled) arrow. Dashed lines indicate the flow of data, with the recipient of the data being indicated with an open arrow. As depicted in Figure 3, a simulation in SIMn proceeds using an event loop, starting at time $t = 0$, that performs the following steps:

1. Check the collected trace for unprocessed WiFi delays that occurred before time t. If WiFi delays exist, t is incremented by the duration of the delay. This is repeated until all WiFi delays that have occurred before t have been processed.
2. Determine any non-WiFi delays that occur at time t and increment t by the duration of the delay.
3. If there are fewer than two aggregate frames (one in transmission and one queued), create an aggregate frame and add it to the queue.

4. Compute the time to transmit the A-MPDU.
5. Determine the fate of each subframe i in the aggregate frame using the Subframe Index Error Rate (SFIER) at time t, rate r and index i. SFIERs are discussed in detail in Section 5.4.1. Failed subframes are rescheduled for retransmission if the retry limit has not been exceeded.

This process repeats until the simulation time is equal to the duration of the collected trace. To create an aggregated frame we first compute the maximum allowed size of the aggregated frame (detailed in Section 5.2.2). Then MPDUs are added to an A-MPDU in a loop until the maximum size is reached. The MPDUs are either new frames arriving from an application or failed MPDUs that are waiting for retransmission. Rescheduled frames are given priority when forming aggregate frames.

To determine the fate of a subframe with index i that needs to be sent at simulated time t with rate configuration r, T-SIMn considers all samples for the rate r and subframe index i in the averaging window (t - window_size/2) ms to (t + window_size/2) ms, from the collected trace. The averaging window should be less than the channel coherence time for the environment so that channel conditions are relatively constant with respect to the frames being used to determine the fate of packets at time t. Channel coherence time depends on many factors (e.g., the speed of movement and channel frequency). In indoor environments at walking speeds channel coherence time is reported to be approximately 200 ms for the 2.4 GHz band [19] and 100 ms for the 5 GHz band [4]. Because all experiments in this paper use the 5 GHz band, we use an averaging window of 200 ms (i.e., $t - 100$ ms to $t + 100$ ms), which corresponds to a 100 ms coherence time. Our evaluation in Section 6 shows that an averaging window of 200 ms produces accurate results in our mobile environment at walking speeds, when performing round-robin trace collection with 1 antenna. However, this parameter can be easily tailored to the environment.

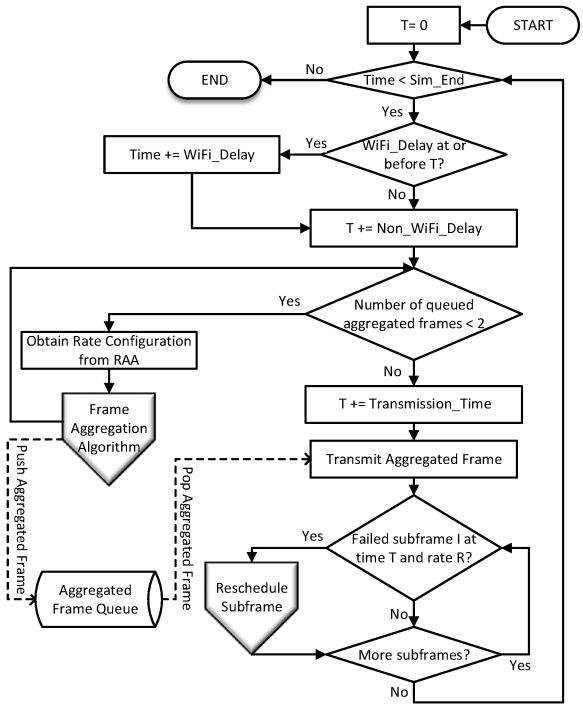

Figure 3: T-SIMn Simulation Flowchart.

3.3 Frame Aggregation Notation

To concisely describe limits on the length of an aggregated frame (i.e., the number of subframes) for a particular rate configuration, we introduce the notation:

$\texttt{FA}_{(\texttt{SIM} \mid \texttt{COL})}$=*maximum number of aggregated subframes*

For example, $\texttt{FA}_{\texttt{SIM}}\texttt{=16}$ means that the *simulator* is allowed to aggregate a maximum of 16 subframes (we omit rate configuration information; it is not needed for our purposes). The number of subframes in a specific aggregated frame may be further limited by the Block-Ack window discussed in detail in Section 5.2.2. Similarly, $\texttt{FA}_{\texttt{COL}}\texttt{=32}$ means that the driver was limited to aggregating a maximum of 32 subframes *during trace collection*. We use the notation $\texttt{FA}_{\texttt{SIM}}\texttt{=MAX}$ and $\texttt{FA}_{\texttt{COL}}\texttt{=MAX}$ to mean that we impose no restrictions, beyond those in the 802.11n standard and the Ath9k driver (i.e., 32 MPDUs), on the aggregated frame length during simulation and trace collection, respectively. Lastly, $\texttt{FA}_{\texttt{SIM}}\texttt{=FA}_{\texttt{COL}}$ means the limits on the aggregated frame are the same during simulation and trace collection.

4. TESTBED

Our test bed is located in cubicles in a large room, as well as some separate offices on a university campus. Our WiFi devices include two desktop computers housing identical TP-Link TL-WDN4800 PCIe cards, an Apple MacBook Pro (Retina, 15-inch, Mid 2012) and an Apple iPhone 6. The TP-Link cards use an Atheros AR9380 chipset, while the MacBook and iPhone use Broadcom BCM4331 and BCM4339, respectively. Although all five devices are dual-band (2.4 and 5 GHz), we limit our experiments to the 5 GHz band because the Apple devices used in some experiments do not support 40 MHz channel bandwidths in the 2.4 GHz spectrum. Except for the iPhone 6 which contains only one antenna, all devices contain three antennas supporting three spatial streams in a 3x3:3 MIMO configuration.

We create an 802.11n Access Point (AP) using hostapd and collect traces while that system sends packets using our modified Ath9k driver. Although the AP could be used as the *sender* or the *receiver*, we use the computer designated as the AP as the *sender* in all experiments. The major advantage of this approach is that there are fewer requirements imposed on the *receiver*, which does not need to be capable of creating an AP. In addition, the *receiver* does not need to use a modified Ath9k driver and, as a result, can be any 802.11n-capable device that runs Iperf3; this includes a wide variety of devices, even an iPhone. We create a network between the AP (*sender*) and a client, which acts as a receiver. To collect a trace the *sender* saturates the link by sending as many 1,470 byte UDP frames as possible using Iperf3. Unless otherwise noted, we aggregate the maximum number of subframes (i.e., $\texttt{FA}_{\texttt{COL}}\texttt{=MAX}$).

We use the second desktop as a *receiver* in the stationary experiments. It is located less than 1 meter from the AP with line of sight so we can collect error-free traces needed for some experiments. The MacBook and iPhone are used for mobile experiments. When it is necessary to minimize the amount of uncontrolled WiFi interference, we use channels in the 5 GHz band that are unused by other APs in the vicinity. We monitor all channels for interference using an AirMagnet Spectrum XT spectrum analyzer. For generating controlled non-WiFi interference we use an RF Explorer Handheld Signal Generator (RFE6GEN) that we control programmatically using a USB connection.

5. T-SIMN DETAILS AND EVALUATION

5.1 Experimental Methodologies

To evaluate the simulator component of the T-SIMn we use a technique that minimizes our reliance on the trace collection methodology. We conduct an experiment using a constant rate configuration which produces a constant rate configuration trace (i.e., the trace collection is also an experiment). We then use SIMn to simulate a constant rate configuration experiment (for the same rate configuration) using the collected trace. Because the simulated experiment uses the same rate configuration as the conducted experiment, simulated throughput should match throughput obtained during the conducted experiment if the simulator is accurate. There are two major advantages to this methodology: (1) It does not require experiments to be repeatable since the experiment produces a trace that is used by SIMn to simulate an experiment with exactly the same conditions and environment as the conducted experiment (i.e., the conducted experiment *is* trace collection). (2) Constant rate traces provide many samples in each averaging window, which allows us to study the accuracy of SIMn without being limited by the collected trace.

Together these properties allow us to expect, and check for, close matches between experimental and simulated throughput when evaluating SIMn. Recall that the simulation *is not simply a trace playback* as discussed in Section 3.2. Our plots include 95% confidence intervals and we consider a match to be obtained if we have overlapping confidence intervals for experimental and simulated throughput over each window of time. Note that in some plots the confidence intervals are so tight that they may not be visible. In this section, all experiments are conducted on channels free of any uncontrolled inference.

5.2 Computing Transmission Duration

Determining the time spent transmitting a frame depends on the combination of physical layer features being used (i.e., the rate configuration) and, at the MAC layer, the number of frames and method used for frame aggregation. We now describe relevant simulator details and evaluate the fidelity of the simulator with respect to these 802.11n features.

5.2.1 Physical Layer Features

The 802.11n protocol introduces many new physical layer features, namely Multiple Spatial Streams (SSs), Short Guard Intervals (SGIs), Channel Bonding and Dual-Band support. Despite being numerous, these features are relatively simple to simulate because how they influence the transmission time is either specified in or can be easily determined from the 802.11n standard. We now evaluate SIMn's ability to accurately replicate the timings, and consequently the throughput, of these 802.11n physical layer features.

Experiment Setup and Description: We create a network between the Access Point (AP) (*sender*) and the desktop client (*receiver*). We use the stationary client because it can reliably obtain error-free traces due to its close proximity and line of sight. We collect 100 second traces for the rate configurations `1S-I4-LG-20M=39`, `1S-I4-SG-40M=90`, `2S-I4-SG-40M=180` and `3S-I4-SG-40M=270`. We choose these rate configurations because they cover both long and short Guard Intervals (GIs); 20 and 40 MHz Channel Bandwidths (CBs); as well as one, two and three spatial streams. A Modulation and Coding Scheme Index (MCS index) of 4 is chosen because it is the highest index with which we could reliably obtain error-free traces. We use highest rates possible because discrepancies between experimental and simulated throughput are more likely to be seen at high rates than low rates. In this experiment, we aggregate as many MPDUs as possible during trace collection and simulation.

Experiment Results: In Figure 4, we plot pairs of throughput measurements, simulated and experimental, for each of the collected rate configurations. For all four rates, the simulated throughput tightly matches the experimental throughput. This suggests that SIMn accurately handles rate configurations using different physical layer transmission features. In other words, SIMn accurately calculates the transmission time of a frame (including ACK and DCF timing) for the combinations of the physical layer features shown. Note that we have tested other combinations (not included here) to confirm that the simulator matches the expected experimental throughput.

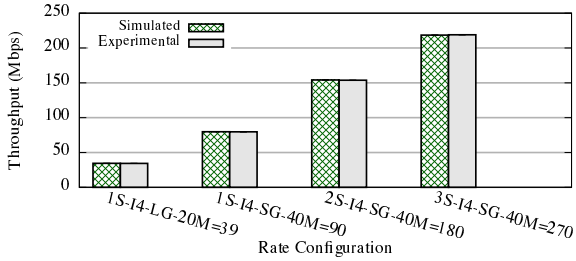

Figure 4: Fidelity of PHY layer features simulation

5.2.2 MAC Layer Features

To understand how to simulate frame aggregation, we first describe the factors that affect the length of an A-MPDU:

(a) There is an air time limit of 4 ms in the 5 GHz ISM band that prevents a single device from monopolizing the channel. This limit means that when using slower rate configurations (i.e., lower Physical Layer Data Rates (PLDRs)), aggregated frame length is limited to the number of frames that can be transmitted in 4 ms. The Ath9k driver applies the same restriction for the 2.4 GHz band.

(b) The Block-Ack Window (BAW) also plays a major role in the length of aggregated frames, as the sequence numbers of subframes can be offset by at most 64 from the starting sequence number in the BAW.

(c) There is a 64 KB limit on the size of the 802.11 PLCP. For example, if using 1,500 byte frames, a maximum of 43 frames may be aggregated.

(d) Implementation specific requirements may also limit the length of aggregated frames. For instance, although this is not dictated by the 802.11n standard, the Ath9k driver used in this paper imposes a limit of 32 subframes in an aggregate frame. This is done so that another aggregated frame can be constructed and queued while the previous frame is being transmitted. A limit of 32 MPDUs in each A-MPDU means that two aggregated frames can be created within the 64-bit BAW, one in transmission and one queued.

(e) Management frames must be transmitted as individual frames. When a time sensitive management frame, such as a beacon frame, is queued the Ath9k driver will stop aggregating subsequent frames. This allows the management frame to begin transmission sooner than if a long aggregated frame were ahead of it in the transmission queue.

Determining the time spent transmitting the frame will depend on the number of frames that have been aggregated and the time required to transmit those subframes, which can be determined from the 802.11n specification. As mentioned previously, trace collection is performed by aggregating as many subframes as possible (i.e., $\texttt{FA}_{\texttt{COL}}\texttt{=MAX}$, which is $\texttt{FA}_{\texttt{COL}}\texttt{=32}$ in Ath9k). We do not collect $\texttt{FA}_{\texttt{COL}}\texttt{=1}$, $\texttt{FA}_{\texttt{COL}}\texttt{=2}$, ..., $\texttt{FA}_{\texttt{COL}}\texttt{=MAX-1}$. However, the simulator will need to accurately simulate cases where $\texttt{FA}_{\texttt{SIM}} \leq \texttt{FA}_{\texttt{COL}}\texttt{=MAX}$ due to the reasons listed above, in addition to permitting different frame aggregation algorithms to be implemented in the simulator (recall that a goal of this work is to enable the fair comparison of different frame aggregation algorithms). Being able to accurately simulate A-MPDUs of any length *using frames collected using only A-MPDUs of maximum length* is they key insight and critical requirement to enable trace-based simulation of 802.11n networks. Therefore, we now evaluate SIMn's accuracy when simulating the throughput of frames aggregated with fewer subframes during simulation than were obtained during trace collection.

Experiment Setup and Description: We create a network between the AP (*sender*) and desktop computer (*receiver*), which is located less than 1 meter away in order to reliably obtain error-free traces (we consider errors in Section 5.4). For all experiments, the *sender* is set to a constant rate configuration of `2S-I4-SG-40M=180`. We collect a 100 second trace with $\texttt{FA}_{\texttt{COL}}\texttt{=MAX}$, as this is the aggregation limit that we will typically use for trace collection. We then conduct 100 second experiments with Frame Aggregation (FA) limited to 32, 16, 2 and 1 aggregated frames, which we use as the ground truth. We then simulate constant rate scenarios for the rate configuration `2S-I4-SG-40M=180` with $\texttt{FA}_{\texttt{SIM}}\texttt{=MAX}$, $\texttt{FA}_{\texttt{SIM}}\texttt{=16}$, $\texttt{FA}_{\texttt{SIM}}\texttt{=2}$ and $\texttt{FA}_{\texttt{SIM}}\texttt{=1}$, using the $\texttt{FA}_{\texttt{COL}}\texttt{=MAX}$ trace as input to the simulator. We expect simulated throughput for $\texttt{FA}_{\texttt{SIM}}\texttt{=MAX}$, $\texttt{FA}_{\texttt{SIM}}\texttt{=16}$, $\texttt{FA}_{\texttt{SIM}}\texttt{=2}$ and $\texttt{FA}_{\texttt{SIM}}\texttt{=1}$ to match the throughput obtained directly from the experiments run with FA limited to 32, 16, 2 and 1 aggregated frames, respectively.

Experiment Results: In Figure 5, we plot pairs of simulated and experimental throughput measurements for each of the frame aggregation configurations. For all pairs of simulated and experimental throughput we see a close match, which suggests that SIMn can accurately simulate shorter aggregated frames from traces with $\texttt{FA}_{\texttt{COL}}\texttt{=MAX}$, in an error-free environment. In Section 6 we demonstrate that SIMn is also accurate in environments that are not free from errors (e.g., including uncontrolled environments). As a result, we are able to collect traces with $\texttt{FA}_{\texttt{COL}}\texttt{=MAX}$ but to simulate cases where frames of shorter lengths are used.

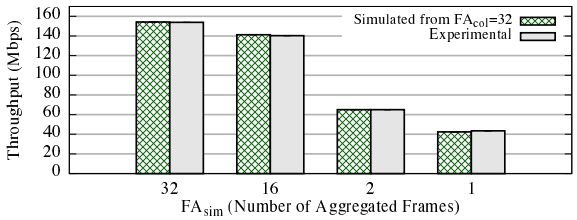

Figure 5: Simulating Shorter Aggregated Frames

5.3 Determining Channel Access

Since the 802.11 standard implements CSMA/CA, channel access is a crucial factor in determining throughput for an experiment. When a WiFi device performs channel sensing and a WiFi or non-WiFi signal is detected the transmission has to be postponed, resulting in reduced throughput. It is critical that T-SIMn measures this delay *while conducting trace collection* and accurately simulates it in order to produce realistic throughputs. Such delays can be caused by changing channel conditions and it is the ability to capture these delays that makes trace-based simulation so attractive. *Channel conditions are captured in traces, rather than simulated using computationally intensive and inaccurate models.* In T-SIMn, we compute both WiFi and non-WiFi delay using Cycle Counter Information (CCI) available on the Atheros device.

We use four counters available on the Atheros chipset which are: *tx_cycles*, which counts the number of cycles that the chip spends performing transmissions (outbound); *rx_cycles*, which counts clock cycles spent receiving (inbound); *busy_cycles*, which measures the number of cycles that the channel was busy performing transmission, receiving WiFi frames or due to non-WiFi noise; and *total_cycles*, which counts the total number of cycles for transmission, including those spent busy (i.e., waiting). We modify the driver to include cycle counts for each aggregated frame in the collected trace and compute delay as follows:

$$delay = actual\ duration - expected\ duration \quad (1)$$

The *expected duration* is determined from the 802.11n standard and includes time for transmission and the Distributed Coordination Function (we use an average backoff of 7.5 μs). The *actual duration* is the time spent transmitting the aggregated frame, as determined using the cycle counters. More details can be found in [2][10].

T-SIMn needs to determine if the source of delay was due to WiFi or non-WiFi interference in order to accurately simulate channel access because they impact delay in different ways. With non-WiFi interference each time the sender attempts to transmit a frame it may experience delay. As a result, the more frequently A-MPDUs are sent the more delay the sender may incur. Since the sender is transmitting a constant stream of packets, the frequency of A-MPDU transmission is affected by the rate configuration and the length of A-MPDUs. The situation is different for WiFi interference because all parties implement the 802.11n standard and cooperate when accessing the channel. As a result, the sender's delay before transmitting an A-MPDU does not depend on the rate configuration or the length of the A-MPDU. It only depends on the amount of time the currently transmitting device occupies the channel being used.

We have developed a simple heuristic to distinguish WiFi from non-WiFi interference. We consider delay to be due to WiFi interference if one or both of the following are true:

1. The time spent transmitting the frame is significantly longer than expected:

$$actual\ tx\ duration - expected\ tx\ duration > threshold$$

$$\frac{tx_cycles}{clock_speed} - data\ tx\ time > threshold$$

This handles the case where the frame is delayed due to outbound traffic on the Access Point (AP), such as transmitting a beacon frame or responding to a probe request. Our current heuristic uses a threshold of 60 μs as it is small enough to catch beacon frames, which are the shortest WiFi frames that we observed.

2. The time spent receiving the ACK for the frame is significantly longer than expected:

$$actual\ rx\ duration - expected\ rx\ duration > threshold$$
$$\frac{rx_cycles}{clock_speed} - ack\ rx\ time > threshold$$

This handles the case where a frame is delayed due to inbound WiFi traffic. Our heuristic uses a threshold of 10 μs because there is little variation in the *actual rx durations* in the absence of WiFi interference.

Although these values provide accurate results in the following evaluation, in future work thresholds may be tuned to improve the accuracy of the heuristic.

If the delay is due to WiFi interference, we simply increment the simulation time by the duration of delay as explained in Section 3.2. To simulate the delay caused by non-WiFi interference, we find the delay incurred when sending each frame (from the trace) and compute the average delay experienced by each frame over the 200 ms time window centered at the current simulation time (i.e., the averaging window used to compute the error rate described in Section 3.2). This average is the expected delay and is then added to the transmission time of the next frame in the simulator. Note that channel sensing does not depend on the rate configuration of the frame to be transmitted, so this computation is independent of the rate configuration.

We now evaluate the accuracy of T-SIMn in presence of WiFi interference, when the simulator must use shorter aggregated frames than those that have been collected (i.e., $\texttt{FA}_{\texttt{SIM}} < \texttt{FA}_{\texttt{COL}}$) due to the size of an A-MPDU being limited by one of the factors explained in Section 5.2.2.

Experiment Setup and Description: We create a network between the AP (*sender*) and a desktop computer client (*receiver*). We collect a trace for the rate configuration `2S-I4-SG-40M=180` with $\texttt{FA}_{\texttt{COL}}$`=MAX`. We then conduct experiments with Frame Aggregation (FA) limited to 2 and 1 aggregated frames. Using the trace collected with $\texttt{FA}_{\texttt{COL}}$`=MAX`, we simulate constant rate scenarios for the rate configuration `2S-I4-SG-40M=180` with $\texttt{FA}_{\texttt{SIM}}$`=2` and $\texttt{FA}_{\texttt{SIM}}$`=1`. This is done to evaluate SIMn's accuracy in simulating shorter frames from traces collected with $\texttt{FA}_{\texttt{COL}}$`=MAX`, in the presence of WiFi interference. We then repeat these simulations but disable the heuristic (i.e., WiFi and non-WiFi delays are not differentiated) to demonstrate how distinguishing the two types of interference improves the accuracy of our simulator.

Experiment Results: Table 1 shows the differences between the throughput obtained with and without the heuristic for $\texttt{FA}_{\texttt{SIM}}$`=2` and $\texttt{FA}_{\texttt{SIM}}$`=1`. In this experiment, the only WiFi interference is from beacon frames generated by the AP. We find that simulation error (i.e., the difference between simulated throughput, with and without the heuristic, and the experimental throughput), is lower when using the heuristic. With the heuristic, simulation error is less than 1% when simulating $\texttt{FA}_{\texttt{SIM}}$`=2` from $\texttt{FA}_{\texttt{COL}}$`=MAX`, and less than 2% when simulating $\texttt{FA}_{\texttt{SIM}}$`=1` from $\texttt{FA}_{\texttt{COL}}$`=MAX`. Without the heuristic, simulation error is roughly 6% and 10% for $\texttt{FA}_{\texttt{SIM}}$`=2` and $\texttt{FA}_{\texttt{SIM}}$`=1`, respectively. These are significant differences, especially when considering that the only WiFi interference is beacon frames (which are relatively short in duration compared to data frames). We expect these differences to be even larger if a third-party WiFi client is added.

Table 1: Simulation Error (% Difference)

$\texttt{FA}_{\texttt{COL}} \rightarrow \texttt{FA}_{\texttt{SIM}}$	With Heuristic	Without Heuristic
$32 \rightarrow 2$	-0.9 %	5.6 %
$32 \rightarrow 1$	1.6 %	10.0 %

5.4 Determining Frame Error Rates

Along with the Physical Layer Data Rate (PLDR) and channel access (delay), the channel frame error rate (FER) is one of the major factors in determining the throughput for an 802.11n network. Errors can be introduced when multiple WiFi or non-WiFi devices transmit at the same time, resulting in a packet collision. Furthermore, errors may be caused by path loss when signal strength is low due to the distance between a *sender* and *receiver* or because of obstacles like walls or furniture that obscure the path between the two devices. We begin by describing the need to consider the subframe index (i.e., the location of the subframe within the A-MPDU) when computing the fate of a subframe in Section 5.4.1. Then, using path loss as an example, we demonstrate that the techniques we have devised provide accurate results (in Section 5.4.2).

5.4.1 Subframe Index Error Rates

In our initial implementation of SIMn, we treated the successes and failures of individual MPDUs (subframes) as samples in the computation of the FER over the specified time window for a particular rate configuration. For example, if we sent 5 aggregated frames, each containing 30 subframes, 15 successful and 15 failing, the FER determined to be 50% because 75 subframes were successfully transmitted and 75 failed. However, we found that simulated throughput did not always closely match experimental throughput, even though the error rates obtained in the simulator were similar to those observed in the experiments. Upon closer inspection, we found that there were significant differences between the average length of aggregated frames in the simulation and in the experiments. In our experiments the error rate of each subframe within multiple A-MPDUs varied with the index of the subframe (i.e., the error rate changed with the location of the subframe within the A-MPDU). Byeon et al. report that subframes transmitted more than 2 ms after the beginning of the transmission of an aggregated frame have a lower probability of being successfully received [6] (i.e., the pattern is that subframes with higher indices have a higher probability of failure). While some of the A-MPDUs we inspected exhibited similar behaviour, we found that the pattern of subframe error rates can differ significantly across different scenarios. As a result, we develop a technique that will work with any pattern, including environments where the pattern changes over time. SIMn now computes individual error rates for each subframe index rather than using an average FER across the entire aggregated frame. We refer to this as a Subframe Index Error Rate (SFIER). We now illustrate the importance of using the SFIERs in obtaining accurate throughput in SIMn.

Experiment Setup and Description: We take advantage of another feature of our trace-based simulator, namely the ability to generate and process synthetic traces to better understand 802.11n networks. We create two synthetic traces with *equal A-MPDU frame error rates* of 41%, but using two different SFIER patterns. The equal FERs are useful in reasoning about the expected outcome. Synthetic traces are

used due to the difficulty in experimentally obtaining traces with the same overall FER with different SFIER patterns. The first trace has a linearly increasing SFIER from 0.025 at index 0 to 0.8 at index 31. The second trace uses the opposite pattern, with the SFIERs decreasing linearly from 0.8 at index 0 to 0.025 at index 31. We use SIMn to simulate the transmission of frames using these two patterns and a rate configuration of `3S-I7-SG-40M=450`. We first treat all subframe indices equally, as in our initial implementation of SIMn. We then repeat the simulations using our current implementation of SIMn that considers each SFIER individually and compare the throughput obtained using these two different approaches.

Experiment Results: Figure 6 plots the throughput obtained for the two synthetic traces. The bars on the left show the results obtained using the current implementation with per-SFIERs and those on the right show results obtained using the initial implementation per-SFIERs are not considered (labelled "Original"). As expected, when using the Original implementation the throughput of the Increasing and Decreasing patterns are equal (because they have the same 41% FER. However, when using per-SFIERs, an increasing SFIER pattern results in higher throughput than a decreasing pattern, even though they have the same overall average error rate. The decreasing SFIER pattern results in lower throughput, because failures at the start of an aggregate frame prevent the Block-Ack Window (BAW) from advancing and thus reduces the average length of aggregated frames. These experiments illustrate the impact of considering individual SFIERs and their importance in obtaining accurate simulation results. This is critical because our goal is to use the simulator to evaluate link adaptation and frame aggregation algorithms, which require the correct simulation of phenomenon captured during trace collection.

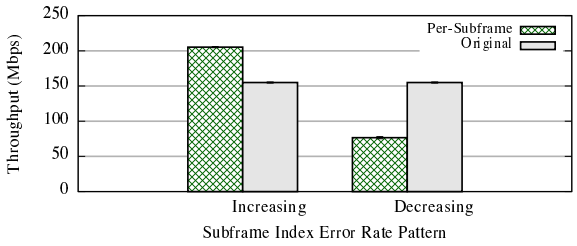

Figure 6: Impact of per-subframe error calculation

5.4.2 *Path Loss*

To evaluate SIMn's ability to handle channel error rate, we use a challenging mobile scenario where channel error rates vary widely due to path loss.

Experiment Setup and Description: We create a network between the Access Point (AP) (*sender*) and a MacBook Pro (MBP). The *sender* transmits for 100 seconds using constant rate configuration of `2S-I4-SG-40M=180` with $\texttt{FA}_{\texttt{COL}}$`=MAX`. We choose this rate configuration because in this mobile scenario, its error rates vary widely. In this experiment, the MBP is carried at walking speed in an office environment where cubicle walls obscure line of sight. We simulate throughput for the rate configuration `2S-I4-SG-40M=180`, using the collected trace as input to the simulator. Note that during the simulation SFIERs must be computed and frames of length shorter than the maximum will be used. This tests SIMn's ability to accurately simulate throughput with fluctuating error rates, different error rates across different subframe indices, and A-MPDUs of different length.

Experiment Results: In Figure 7, we plot pairs of throughput measurements, simulated and experimental, for this scenario. Despite the significant fluctuations during this experiment, there is a close match between simulated and experimental throughput. This suggests that the simulator can accurately determine error rates (including SFIERs) and simulate aggregated frames of different lengths.

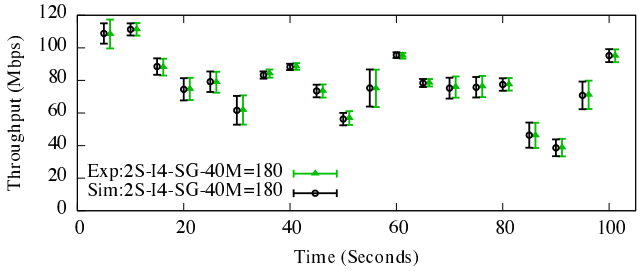

Figure 7: Mobile scenario experiencing path loss

6. EVALUATING OUR FRAMEWORK

In the previous sections, we use constant rate traces to focus our tests on the T-SIMn simulator, SIMn. Although not representative of how trace collection would be done in T-SIMn, that technique is used to ensure the accuracy of SIMn on its own. In this section, we evaluate T-SIMn as a whole, using round-robin trace collection in conditions that are representative of those in which WiFi is used.

In order to evaluate the T-SIMn framework, we use an evaluation methodology similar to that used to evaluate T-RATE [2]. That is, we conduct an experiment (which produces a trace) using a round-robin ordering of rate configurations and then in SIMn we conduct a simulation using a round-robin ordering that differs from the experiment. This experiment is designed as a means for evaluating the accuracy of the T-SIMn framework. The intuition behind this methodology is that in both the experiments and the simulation each rate will be used to send the same number of frames using each rate. Therefore, the average throughput obtained over an interval in time from the experiment should be matched by the average throughput obtained from the simulator. This will only be true if, despite not having sent a frame with rate configuration R at time t, the simulator can accurately determine the probability that the frame would be successfully sent by computing the average SFIER over the channel coherence window. Additionally, different orderings means it is extremely unlikely that the number of frames aggregated in the simulator at time t will match the number of aggregated frames collected in the trace at time t. In contrast to T-RATE, we cannot cycle through all of the available 802.11n rates (96 rates for our 3 antenna devices would take about 300 ms) because the time required to perform enough rounds to accurately compute average error rates would exceed the channel coherence time.

We instead limit our evaluation to one antenna (1-Spatial Stream (SS)) devices which includes most cell phones, tablets and other small devices. Using Long Guard Interval (LGI), Short Guard Interval (SGI), 20 and 40 MHz Channel Bandwidth (CB) results in a total of 32 rate configurations.

During trace collection, rates are grouped by a combination of the Guard Interval (GI) and the CB. Therefore, rates are sampled in the following group order `LGI-20MHz`, `SGI-`

20MHz, LGI-40MHz, SGI-40MHz. Within each group, rates are sampled in order from the lowest Modulation and Coding Scheme Index (MCS index) to the highest (i.e., MCS index $0, 1, ..., 6, 7$). We simulate round-robin in the reverse group order (i.e., SGI-40MHz, LGI-40MHz, SGI-20MHz, LGI-20MHz) and from the highest MCS index to the lowest (i.e., MCS index $7, 6, ..., 1, 0$).

Experiment Setup and Description: We create a network between the AP (*sender*) and an iPhone 6 (*receiver*). The sender is configured to sample all 32 1-SS 802.11n rate configuration, which is the entire set of rates supported by the iPhone 6. The experiment is conducted by using a mix of carrying the iPhone at walking speed and standing still in an office environment as described in Section 4.

Experiment Results: In Figure 8, we plot simulated and experimental throughput. We find that simulated throughput matches experimental throughput except for the points at times $t = 20$, $t = 60$ and $t = 100$. The largest difference is observed at $t = 60$, where average simulated throughput is roughly 11% higher than average experimental throughput. Initially, we thought that this was due to inaccuracy in SIMn. However, upon closer investigation, we realized that the simulator is in fact accurate and that the problem was with the methodology used to evaluate the accuracy of the framework. Our assumption that simulating round-robin in a different order would result in each rate configuration being used for the same proportion of time is not guaranteed in 802.11n networks, unlike a similar evaluation we conducted for 802.11g networks [2]. In the next section, we investigate and explain why the order in which rates are used in a round-robin fashion impacts throughput.

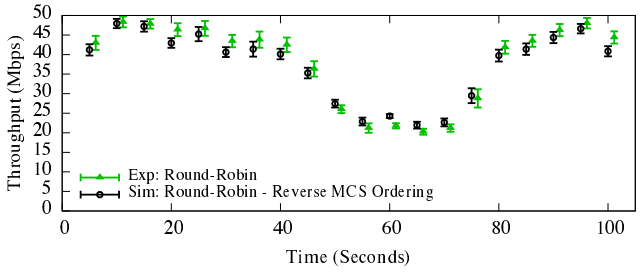

Figure 8: Simulating Reverse MCS Ordering

6.1 Effect of Rate Configuration Ordering

In 802.11n networks, the fate of one frame can impact the number of frames that can be aggregated in the next frame due to the Block-Ack Window (BAW). Failed aggregated frames or subframes limit how far forward the BAW can be advanced. Recall from Figure 1 that the number of subframes being aggregated has a significant impact on throughput, with longer aggregated frames leading to higher potential throughput. Recall that in the previous section, rates were sampled in the group order LGI-20MHz, SGI-20MHz, LGI-40MHz, SGI-40MHz. Additionally, rates are sampled in order from the lowest Modulation and Coding Scheme Index (MCS index) to the highest (i.e., MCS index $0, 1, ..., 6, 7$). This means that the most robust rates in each group are sampled first and the least robust rates are sampled last. We have examined the data shown in Figure 8 in detail and found that at times $t = 50$ to 70, the reverse ordering performed during simulation leads to longer aggregated frames on average (a mean of 15.2 subframes in each aggregated frame during simulation compared to 14.6 in the experiment). This results in slightly higher simulated throughput during this period. Although there is a match in simulated and experimental throughput during some portions of this time interval (i.e., overlapping CIs), the simulated throughputs are visibly lower for most times t. Simulating longer frames than those that were collected may also lead to some inaccuracies because of insufficient samples in the collected trace for frames of the length being simulated.

As a result, we have had to revise our understanding and now expect different round-robin orderings to result in different throughput, unless the behavior of the Block-Ack Window (BAW) advancement and consequently Frame Aggregation (FA) is the same during trace collection and simulation. To test this hypothesis, we construct a new ordering to use in the simulation that preserves the property that the most robust rates in each group are used first and the least robust rates are used last. The simulation still uses rate groups in the reverse order from the order used when the trace is collected (i.e., simulating SGI-40MHz, LGI-40MHz, SGI-20MHz and LGI-20MHz. However, within each group, we now use rates in order from the lowest Modulation and Coding Scheme Index (MCS index) to the highest (i.e., the same order used during trace collection). We simulate a round-robin ordering with this new reverse group order and show simulated and experimental throughput in Figure 9. As the graph shows, the simulated throughput now closely matches that obtained experimentally (all pairs of confidence intervals overlap). Note that this property does not limit SIMn to simulating only certain orderings of rate configurations. It is only required for this evaluation of the accuracy of T-SIMn because we are trying to devise a methodology where the throughput of the simulator should match that of the experiment. Now that we are aware of this property, we will use the reverse group ordering in the following section, where we evaluate T-SIMn in an uncontrolled environment.

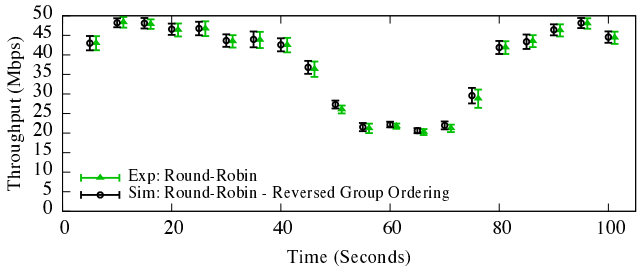

Figure 9: Simulating Reverse Group Ordering

6.2 Uncontrolled Environment

Up to this point, all experiments are performed with no neighboring Access Points (APs). We now move to a different 5 GHz channel that is in use by the university's WiFi network to evaluate T-SIMn in conditions that are typical for a shared university WiFi network. This includes interference from many third-party WiFi clients and APs.

Experiment Setup and Description: Similarly to the previous section, we create a network between the AP (*sender*) and an iPhone 6 (*receiver*). However, unlike previous experiments, we now configure the AP to use a channel occupied by one of the university's APs with the highest signal strength. As mentioned previously, we use a 5 GHz network because the iPhone does not support 40 MHz Channel

Bandwidths (CBs) in the 2.4 GHz spectrum. Note that if we had used the 2.4 GHz band, thus limiting trace collection to 20 MHz rate configurations, we would have obtained twice as many samples in each averaging window which should only improve accuracy. As in previous experiments, we sample all 1-Spatial Stream (SS) rate configurations. We collect a 100 second trace with `FA_COL=MAX` to test T-SIMn in an uncontrolled environment. This scenario is comprised of a mix of carrying the iPhone at walking speed and standing still in an office and hallway environment as explained in Section 4.

Experiment Results: In Figure 10, we plot pairs of throughput measurements, simulated and experimental, for the uncontrolled experiment. We find that while the *receiver* is stationary from $t = 60$ to 100 there is significantly more fluctuation in throughput when compared to Figure 8 from $t = 10$ to 40 in Figure 9. This is due to the third-party traffic on the shared channels. The close matches in throughput suggest that T-SIMn is accurately capturing and simulating third-party traffic and that it can handle conditions that are representative of those in which WiFi devices are used.

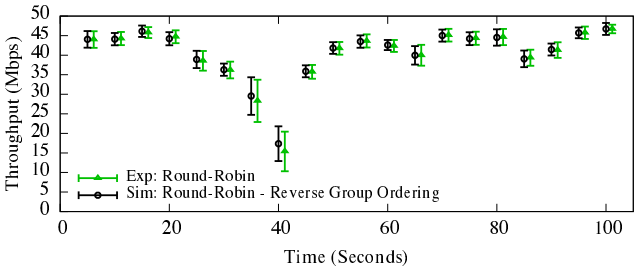

Figure 10: Uncontrolled, Mobile Scenario

7. CONCLUSIONS

In this paper, we design a trace-based simulation framework for 802.11n networks, called T-SIMn. We find that careful consideration of all factors that affect throughput such as 802.11n transmission features, channel access, and channel error rate is necessary to accurately simulate this standard. More specifically, we show that accurate handling of 802.11n frame aggregation is a key in obtaining realistic and high fidelity results. We demonstrate that SIMn accurately simulates these factors by comparing the simulator results with empirical measurements.

We demonstrate that the T-SIMn framework can be used with 1 antenna devices, including most smart phones and tablets. With so many devices being limited to 1 Spatial Stream (SS), we believe that T-SIMn is a valuable tool for studying such widely used devices and that the groundwork has been laid for simulating more SSs (i.e., support for these rates has been implemented and tested in SIMn).

We have made some traces available [1] and intend to add more traces and T-SIMn to this repository. We also plan to use T-SIMn to evaluate link adaptation and frame aggregation algorithms using a wide variety of environments.

8. ACKNOWLEDGMENTS

Funding for this project was provided in part by a Natural Sciences and Engineering Research Council (NSERC) of Canada Discovery Grant and a Discovery Accelerator Supplement in addition to an NSERC Graduate Scholarship. The authors thank Martin Karsten, Srinivasan Keshav and the anonymous reviewers for their helpful feedback on previous revisions of this work.

9. REFERENCES

[1] Sample traces. https://cs.uwaterloo.ca/~brecht/t-simn.
[2] A. Abedi and T. Brecht. T-RATE: A framework for the trace-driven evaluation of 802.11 rate adaptation algorithms. In *MASCOTS*, 2014.
[3] A. Abedi, A. Heard, and T. Brecht. Conducting repeatable experiments and fair comparisons using 802.11n MIMO networks. *ACM SIGOPS*, 49(1):41–50, 2015.
[4] J. G. Andrews, A. Ghosh, and R. Muhamed. *Fundamentals of WiMAX: understanding broadband wireless networking*. Pearson Education, 2007.
[5] R. Burchfield, E. Nourbakhsh, J. Dix, K. Sahu, S. Venkatesan, and R. Prakash. RF in the jungle: Effect of environment assumptions on wireless experiment repeatability. In *ICC*, 2009.
[6] S. Byeon, K. Yoon, O. Lee, S. Choi, W. Cho, and S. Oh. MoFA: Mobility-aware frame aggregation in Wi-Fi. In *CoNEXT*, 2014.
[7] L. Deek, E. Garcia-Villegas, E. Belding, S.-J. Lee, and K. Almeroth. Intelligent channel bonding in 802.11n WLANs. *IEEE Transactions on Mobile Computing*, 13(6):1242–1255, 2014.
[8] L. Deek, E. Garcia-Villegas, E. Belding, S.-J. Lee, and K. Almeroth. A practical framework for 802.11 MIMO rate adaptation. *Computer Networks*, 2015.
[9] D. Halperin, W. Hu, A. Sheth, and D. Wetherall. Predictable 802.11 packet delivery from wireless channel measurements. *ACM SIGCOMM Computer Communication Review*, 41(4):159–170, 2011.
[10] A. Heard. T-SIMn: Towards a Framework for the Trace-Based Simulation of 802.11n Networks. Master's thesis, University of Waterloo, 2016.
[11] P. Huang, X. Yang, and L. Xiao. Adaptive channel bonding in multicarrier wireless networks. In *MobiHoc*, 2013.
[12] G. Judd and P. Steenkiste. Repeatable and realistic wireless experimentation through physical emulation. *ACM SIGCOMM Computer Communication Review*, 2004.
[13] G. Judd and P. Steenkiste. Using emulation to understand and improve wireless networks and applications. In *NSDI*, 2005.
[14] D. Kotz, C. Newport, R. S. Gray, J. Liu, Y. Yuan, and C. Elliott. Experimental evaluation of wireless simulation assumptions. In *MSWiM*, 2004.
[15] L. Kriara and M. K. Marina. SampleLite: A hybrid approach to 802.11n link adaptation. *ACM SIGCOMM Computer Communication Review*, 45(2):4–13, 2015.
[16] M. Lacage and T. R. Henderson. Yet another network simulator. In *Proceeding from the 2006 Workshop on ns-2: The IP Network Simulator*. ACM, 2006.
[17] V. Lenders and M. Martonosi. Repeatable and realistic experimentation in mobile wireless networks. *IEEE Transactions on Mobile Computing*, 2009.
[18] Scalable Network Technologies, Inc. QualNet, 2014. http://web.scalable-networks.com/qualnet.
[19] W.-L. Shen, Y.-C. Tung, K.-C. Lee, K. C.-J. Lin, S. Gollakota, D. Katabi, and M.-S. Chen. Rate adaptation for 802.11 multiuser MIMO networks. In *Mobicom*, 2012.
[20] M. Stoffers and G. Riley. Comparing the ns-3 propagation models. In *MASCOTS*, pages 61–67. IEEE, 2012.
[21] R. Technology. SteelCentral Riverbed Modeler, 2016. http://www.riverbed.com/products/steelcentral/steelcentral-riverbed-modeler.html.
[22] P. Teymoori, A. Dadlani, K. Sohraby, and K. Kim. An optimal packet aggregation scheme in delay-constrained IEEE 802.11n WLANs. In *WiCOM*, 2012.
[23] Z. Zhao, F. Zhang, S. Guo, X.-Y. Li, and J. Han. RainbowRate: MIMO rate adaptation in 802.11n WiLD links. In *IPCCC*, 2014.
[24] J. Zhou, Z. Ji, and R. Bagrodia. TWINE: A hybrid emulation testbed for wireless networks and applications. In *INFOCOM*, 2006.