Computer Communications 117 (2018) 116–132

Contents lists available at ScienceDirect

Computer Communications

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

T-SIMn: A trace collection and simulation framework for 802.11n networks

Ali Abedi*, Tim Brecht, Andrew Heard

David R. Cheriton School of Computer Science, University of Waterloo, Canada

ARTICLE INFO

Article history:
Available online 17 August 2017

Keywords:
Trace-based simulation
Performance evaluation
802.11
WiFi
Trace collection

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. In order to accurately estimate the fate of frames that could have been sent at any rate, traces collected for use in trace-based simulators, like T-SIMn, require a sufficiently large number of samples to be collected using each different rate in a relatively short period of time. In this paper, we devise two novel techniques for collecting and processing traces for 802.11n networks that incorporate Frame Aggregation (FA). The first technique, called direct measurement, samples all rates while aggregating the maximum number of possible frames for each sample. This approach is attractive because frame error rates (FERs) may vary with the position of the frame within the aggregated frame and this techniques directly captures the fate of each subframe. However, the length of the aggregated frames limits this approach to smaller numbers of rates, making it unusable for devices with 3 antennas (e.g., 96 rates). As a result, we also devise second technique that collects traces with frame aggregation turned off, permitting a larger number of rates to be sampled within the same period of time. This approach, called inferred measurement, infers the FER of each subframe using models derived from calibration traces combined with a new measure of changing channel conditions we call the channel dynamic indicator (CDI). Using the direct measurement methodology, we evaluate the T-SIMn framework 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 several scenarios and demonstrate the fidelity of SIMn by uncovering problems with our initial evaluation methodology. We then demonstrate that our inferred measurement technique permits us to collect traces that sample all 96 rates in a 3x3:3 802.11n MIMO systems. These traces are then used to accurately simulate transmissions in environments with highly variable channel conditions that include mobility and multiple sources of interference.

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.

Crown Copyright © 2017 Published by Elsevier B.V. All rights reserved.

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 transmission rates 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 [1] that uses traces to capture environmental conditions, rather than models, to simulate

* Corresponding author.
E-mail addresses: ali.abedi@uwaterloo.ca, ali.a.abedi@gmail.com (A. Abedi), brecht@cs.uwaterloo.ca (T. Brecht), asheard@uwaterloo.ca (A. Heard).

http://dx.doi.org/10.1016/j.comcom.2017.08.008
0140-3664/Crown Copyright © 2017 Published by Elsevier B.V. All rights reserved.

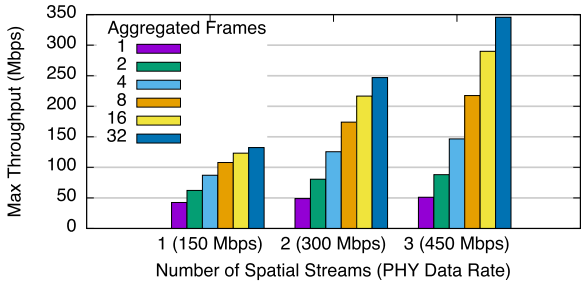


Fig. 1. FA: Maximum theoretical throughput.

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 transmission rates 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 layer frames to be combined into a large physical layer frame so that they can be transmitted and acknowledged as one aggregated frame, which results in the channel being used more efficiently.

To demonstrate the importance of FA, Fig. 1 shows the maximum theoretical throughput obtained using the highest physical layer transmission rate 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.

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 [2] (which studies physical layer configurations such as rate adaptation [3,4] and channel bandwidth adaptation [5,6]) and frame aggregation algorithms [7,8]. We refer to link adaptation and frame aggregation algorithms collectively as *optimization algorithms*.

The contributions of this paper are:

- We design, implement and evaluate a trace-based simulation framework (T-SIMn) for realistic and repeatable performance evaluations of 802.11n networks.
- We design and evaluate a trace collection methodology (called *direct measurement*), which is simple and easy to use but suitable only for devices with a smaller number of transmission rates (e.g., the vast majority of cellphones and tablets that have only one antenna). Using an iPhone as the receiving device we show that collecting traces using direct measurement combined with our prototype implementation of T-SIMn produce highly accurate simulation results.
- We design and evaluate a trace collection methodology (called *indirect measurement*), which is more complex to use but is suitable for devices with many more transmission rates (e.g., some desktop systems with three antennas supporting 96 rates). We demonstrate that indirect measurement can be combined with T-SIMn to obtain highly accurate simulations of a device with 96 rates.
- While designing the indirect measurement methodology we developed a new technique for estimating the frame error rate of subframes within an aggregated frame that takes into account channel dynamics. We first show that changes in the Received Signal Strength Indicator (RSSI) of ACKs can be used to infer the channel dynamics which in conjunction with the delivery ratio of the first MAC Protocol DATA Unit (MPDU) can be used to accurately estimate the fate of other subframes in an aggregated MPDU (A-MPDU). We then develop a model that can be used with the T-SIMn simulator and demonstrate that the model is accurate.

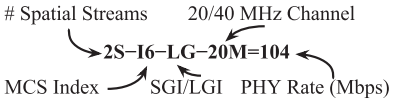


Fig. 2. Rate notation.

The paper begins by providing some necessary background and describing related work in Section 2. We then present an overview of the T-SIMn framework in Section 3, followed by a description of the test bed used in this study (in Section 4). In Section 5, we describe the components of the T-SIMn framework and evaluate the accuracy of these components. We then evaluate the framework as a whole in Section 6, by utilizing traces collected using an iPhone and the direct measurement methodology. In Section 7, we investigate the limitations of the direct measurement methodology when used with devices that support many transmission rates. In Section 8, we propose and evaluate the inferred measurement methodology which is designed to be used in situations where the direct trace collection technique is unable to sample enough rates within the required period of time. Finally, conclusions are presented in Section 9.

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, Fig. 2 introduces notation used for describing a rate configuration (sometimes simply referred to as a rate).

The first two characters represent the number of spatial streams (in the example 2S stands for two spatial streams). The next two characters represent the MCS index (I6 means index 6). This is followed by information about whether a Long Guard Interval (LGI) or Short Guard Interval (SGI) are being used (in the example LG means and LGI is used). The next three characters describe whether the channel bandwidth is 20 or 40 MHz (20 MHz in the example). Finally the number after the equal sign indicates the resulting physical rate (104 Mbps in the example).

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-MSDU)*. 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 [9] 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 [10].

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* [11], OMNeT++ [12], OPNET Modeler [13] (renamed, Riverbed Modeler) and QualNet [14]. 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 [15]. These simulators utilize models for signal propagation, error rates and interference and each model must tradeoff computational complexity and accuracy [15]. 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 suffer from practical limitations For instance, OMNeT++ does not support multi-path signal propagation [16], which is an important prerequisite for simulating MIMO transmissions. There is also no support for MIMO in ns-3. 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) that simulates signal propagation [17]. An emulation test bed by Judd and Steenkiste [18] 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 [18,19], 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 [20]. 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 [1] 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 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 [21,22].

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 two trace collection techniques that sample all rate configurations in a round

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

robin fashion. The direct measurement methodology is suitable for studies using devices that support a limited number of transmission rates such as one antenna devices (e.g., many cellphones and tablets). We believe that the direct measurement will be used when possible due to its simplicity. However, for 802.11n devices with a larger number of transmission rates, we design a methodology called inferred measurement. We now explain the common characteristics of both methods and the details of the direct measurement methodology. A detailed description of the inferred measurement methodology is presented in Section 8.

With both trace collection techniques, we modify the device driver so that the rate adaptation algorithm, instead of trying to choose the best rate, cycles through all of the rates supported by the device in a round-robin fashion for the duration of the trace collection. After a frame is transmitted, the *sender* switches to the next rate; wrapping back around after all rates have been sampled. 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 [7] 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 [7]). 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, the direct measurement methodology 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 both methodologies, 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. It may be possible to use other WiFi devices for trace collection. However, we chose the Atheros AR9380 chipset because of the availability of the source code and its support for CCI. Our trace collection requires the modification of the rate adaptation algorithm and, as will be required in Section 7, the frame aggregation algorithm.

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.

Figs. 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 Fig. 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 frames 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 [23] and 100 ms for the 5 GHz band [24]. 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.

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:

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

For example, $\mathtt{FA}_{\mathtt{SIM}}$=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, $\mathtt{FA}_{\mathtt{COL}}$=32 means that the driver was limited to aggregat-

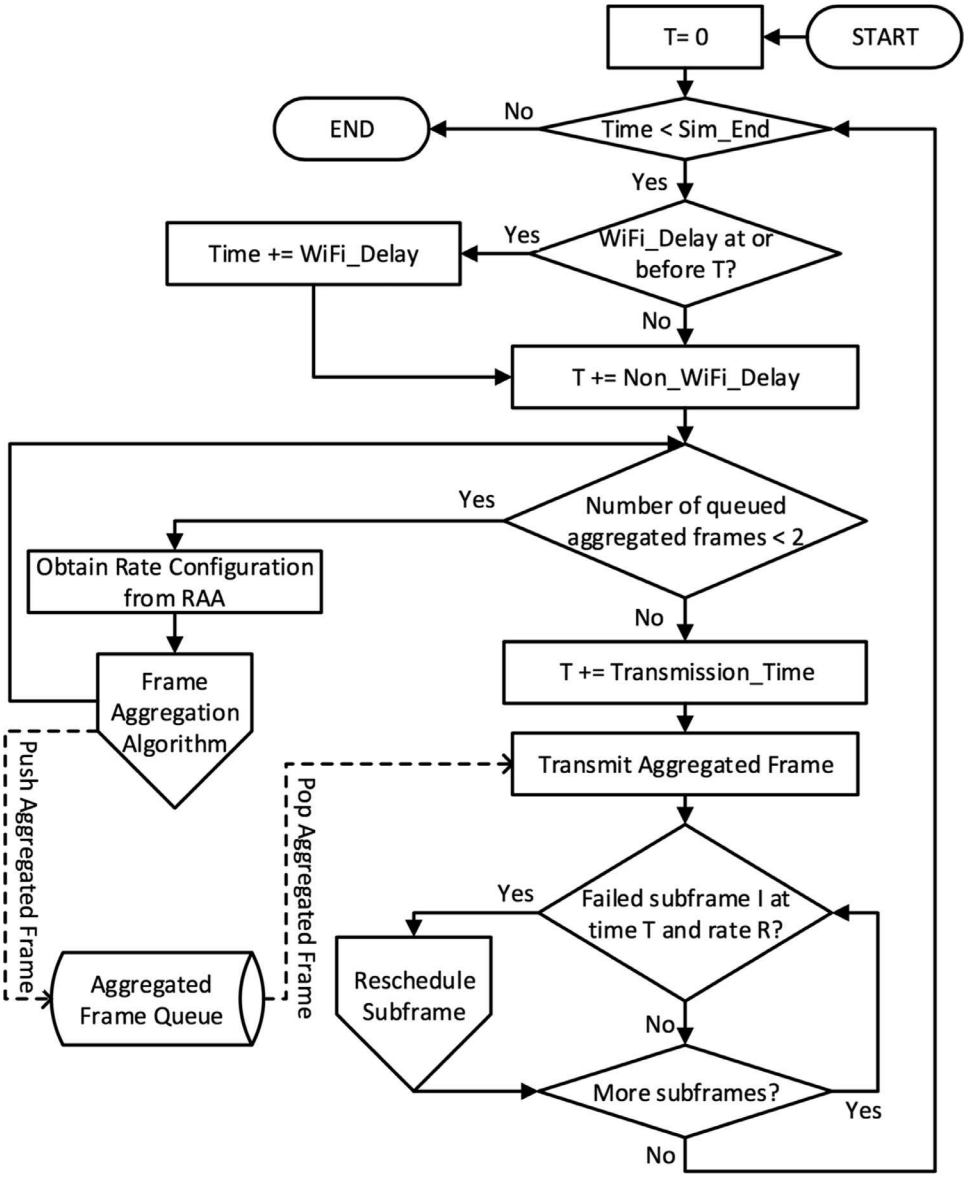


Fig. 3. T-SIMn simulation flowchart.

ing a maximum of 32 subframes *during trace collection*. We use the notation FA_{SIM}=MAX and FA_{COL}=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, FA_{SIM}=FA_{COL} means the limits on the aggregated frame are the same during simulation and trace collection.

4. Test bed

Our test bed is located in cubicles in a large room, as well as some separate offices on a university campus. Fig. 4 shows the floor plan for the cubicles and offices. Four cubicles are located in the area near the labels (AP) and (C) and four are located between the two grey bars (which indicate paneled dividers between the two sets of cubicles). The five smaller rooms are two person offices and the larger room at the top right is a meeting room. Our WiFidevices include two desktop computers housing identical TP-Link TL-WDN4800 PCIe cards, an Apple MacBook Pro-(Retina, 15-inch, Mid 2012), an Apple iPhone 6, and a laptop configured

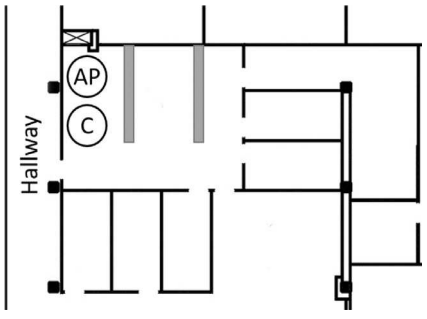


Fig. 4. Floor plan of the test bed environment.

to use a TP-Link TL-WDN4200 dual-band wireless N USB adapter. The TP-Link cards use an Atheros AR9380 chipset, while the MacBook and iPhone use Broadcom BCM4331 and BCM4339, respectively. The USB adapter contains an Ralink RT3573 chipset and uses the rt2800usb (Ralink) device driver. Although all devices are dual-band (2.4 and 5 GHz), several experiments are conducted using the

5 GHz band because the Apple devices used in these experiments do not support 40 MHz channel bandwidths in the 2.4 GHz spectrum. Except for the iPhone 6, which contains only one antenna (i.e., 32 transmission rates), all devices contain three antennas supporting three spatial streams in a 3x3:3 MIMO configuration with 96 transmission rates.

We create an 802.11n Access Point (AP), shown as (AP) in Fig. 4, using hostapd and collect traces while that system sends frames 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*, shown as © in Fig. 4, 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, iPhone, and the USB adapter 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.

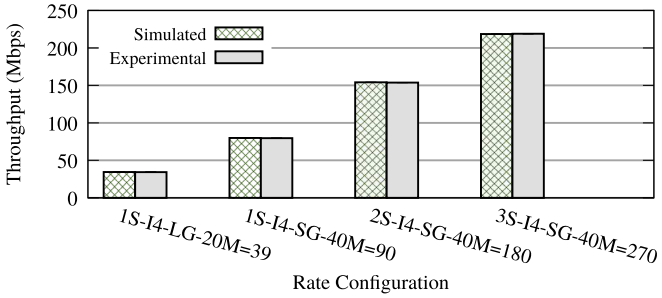


Fig. 5. Fidelity of PHY layer features simulation.

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 Fig. 5, 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.

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., FA_{COL}=MAX, which is FA_{COL}=32 in ath9k). We do not collect FA_{COL}=1, FA_{COL}=2, ..., FA_{COL}=MAX-1. However, the simulator will need to accurately simulate cases where $FA_{SIM} \leq FA_{COL}$=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 FA_{COL}=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 FA_{SIM}=MAX, FA_{SIM}=16, FA_{SIM}=2 and FA_{SIM}=1, using the FA_{COL}=MAX trace as input to the simulator. We expect simulated throughput for FA_{SIM}=MAX, FA_{SIM}=16, FA_{SIM}=2 and FA_{SIM}=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 Fig. 6, 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 FA_{COL}=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 FA_{COL}=MAX but to simulate cases where frames of shorter lengths are used.

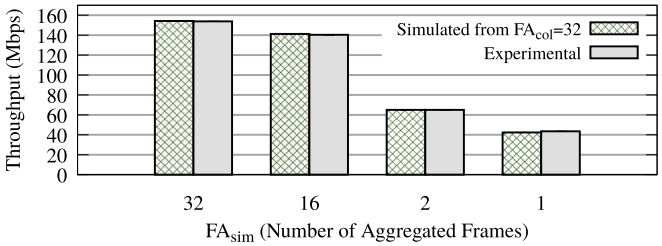


Fig. 6. Simulatingshorter aggregated frames.

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 (out bound); *rx_cycles*, which counts clock cycles spent receiving (in bound); *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 [1,25].

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 frames, 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 out bound 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 in bound 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., $FA_{SIM} < FA_{COL}$) 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 FA_{COL}=MAX. We then conduct experiments with Frame Aggregation (FA) limited to 2 and 1 aggregated frames. Using the trace collected with FA_{COL}=MAX, we simulate constant rate scenarios for the rate configuration 2S-I4-SG-40M=180 with FA_{SIM}=2 and FA_{SIM}=1. This is done to evaluate SIMn's accuracy in simulating shorter frames from traces collected with FA_{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 FA_{SIM}=2 and FA_{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 FA_{SIM}=2 from FA_{COL}=MAX, and less than 2% when simulating FA_{SIM}=1 from FA_{COL}=MAX. Without the heuristic, simulation error is roughly 6% and 10% for FA_{SIM}=2 and FA_{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).

$FA_{COL} \rightarrow FA_{SIM}$	With heuristic	Without heuristic
32 → 2	−0.9%	5.6%
32 → 1	1.6%	10.0%

5.4. Determining frame error rates

Along with the physical layer transmission rate 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-WiFidevices 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 [7] (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

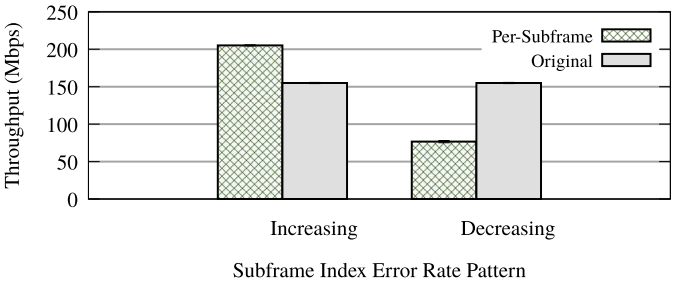


Fig. 7. Impact of per-subframe error calculation.

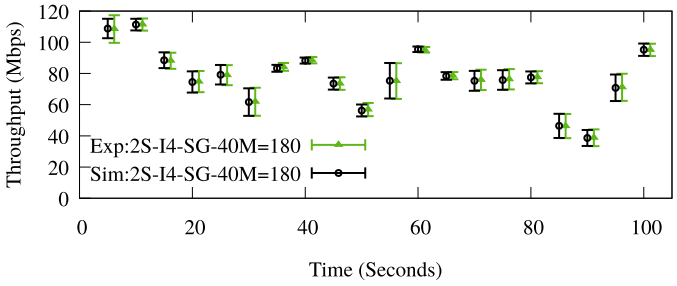


Fig. 8. Mobile scenario experiencing path loss.

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: Fig. 7 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.

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 Mac-Book 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 Fig. 8, 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.

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. *For the experiments in this section, we collect traces using the direct measurement methodology and use an iPhone, which supports 32 transmission rates. We show that this approach can be used with devices that support 32 transmission rates for easy and accurate trace collection. In Section 8, we propose and evaluate a more sophisticated methodology that handles devices that support more transmission rates.*

In order to evaluate the T-SIMn framework, we use an evaluation methodology similar to that used to evaluate T-RATE [1]. 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.

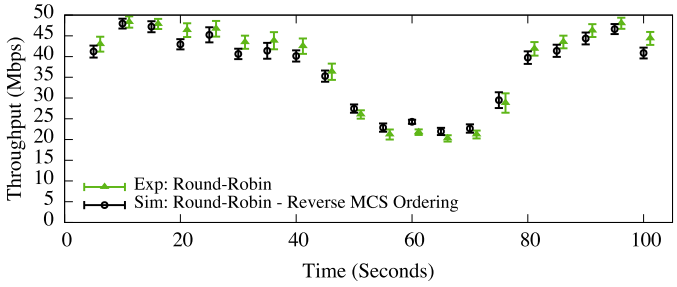


Fig. 9. Simulating reverse MCS ordering.

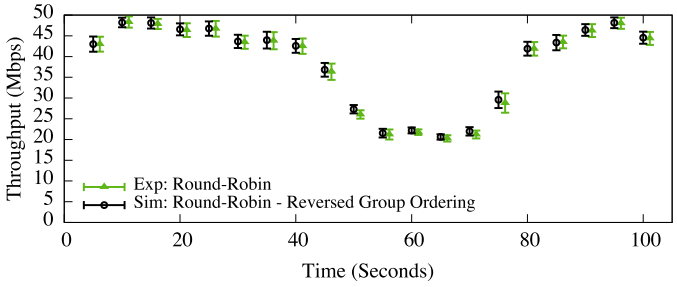


Fig. 10. Simulating reverse group ordering.

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 $0, 1, \ldots, 6, 7$). 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 $7, 6, \ldots, 1, 0$).

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 Fig. 9, we plot simulated and experimental throughput. The experiment starts with the handheld mobile device near the access point. The decrease in throughput from around the 40 second mark to 60 seconds is due to movement away from the access point, while the increase in throughput from 60 seconds until around 80 seconds is due to movement back toward the access point. 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 [1]. In the next section, we investigate and explain why the order in which rates are used in a round-robin fashion impacts throughput.

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 Fig. 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 $0, 1, \ldots, 6, 7$). 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 Fig. 9 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 Fig. 10. 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.

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.

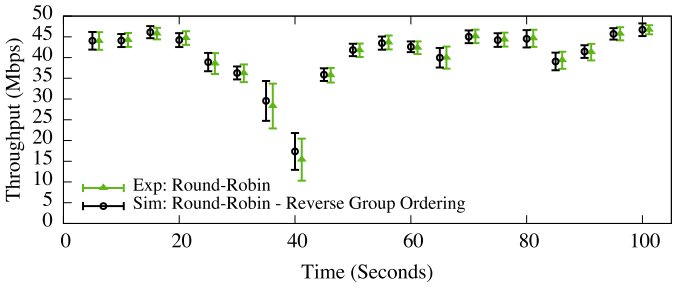


Fig. 11. Uncontrolled, mobile scenario.

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 $\texttt{FA}_{\texttt{COL}}$=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 Fig. 11, 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 Fig. 9 from $t = 10$ to 40 in Fig. 10. 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.

7. Trace collection with many transmission rates

In Section 3.1, we explain that the error rate of the MPDUs within an aggregated frame are not identical. As a result, the direct measurement technique measures the per MPDU error rate by sending the longest possible A-MPDU for each transmission rate. All transmission rates are sampled using a round-robin ordering. As explained in Section 3.1, if we sample all transmission rates within the channel coherence window, all rates experience the same channel conditions. Since the number of transmission rates affects the time required to complete a round of sampling for all, this determines the number of samples that can be collected within the channel coherence window. For 802.11n devices that support many transmission rates, the number of samples collected by the direct measurement technique might be too low to accurately measure the error rate. We refer to this as the *lack of samples problem* for trace collection.

In Section 6, we show that the direct measurement technique can be used for devices that support 32 transmission rates. However, this technique may not be accurate for devices that support more transmission rates. For instance, for a device that supports up to 3 spatial streams (i.e., 96 rates), our measurements have determined that each round of sampling takes about 300 ms to complete when using the direct measurement technique. Recall that the channel coherence time in the 5 GHz spectrum is about 100 ms, as a result, the coherence window is 200 ms (i.e., 100 ms before and after a particular point in time). As a result, each transmission rate only has at most one (per sub-index) sample in a coherence window, which may not be enough to capture fast changing channel conditions. While we have shown the direct measurement methodology is accurate for 32 rates, the maximum number of rates that this methodology can support depends on the required accuracy for a particular study and is a topic of future work.

However, in the rest of this section, we study the lack of samples problem with the goal of alleviating the shortcomings of the direct measurement technique. We show that a relatively intuitive approach that one might expect to work can not be used and argue that more sophisticated techniques are required. In Section 8, we propose a novel methodology that can be used to obtain a sufficient number of samples with devices supporting a larger number of rates (i.e., three antennas with 96 rates).

7.1. Dynamic transmission rate elimination

One approach to increasing the number of samples within the coherence window is to dynamically eliminate "trivially predictable" transmission rates when collecting a trace using a round robin ordering. The intuition is that if a particular transmission rate fails to successfully transmit a frame, faster transmission rates (which use less redundancy) will not be able to transmit the frame either (i.e., the FER will be 1). Similarly, if a particular transmission rate is successful, sampling slower transmission rates is not necessary as they are even more robust and will be successful (i.e., the FER will be 0). Therefore, sampling these transmission rates with known expected FER is wasteful and sampling all rates might be considered unnecessary and "oversampling" some rates. As a result, the round robin trace collection mechanism might be able to skip many rates and collect a larger number of more useful samples by dynamically eliminating some rates.

This heuristic could solve the lack of samples problem if a large enough portion of the rates experience an error rate sufficiently close to 0 or 1. These rates that are trivially predictable over some short periods of time (i.e., *trivial rates*) can be temporarily and dynamically excluded or included by the heuristic. Note that the set of trivial rates might change over time as the channel conditions change. To evaluate the potential of such a heuristic, we initially designed an experiment to examine the number of *non-trivial rates*, with $\text{FER} \neq 0$ or $\text{FER} \neq 1$, over a one second window. However, we soften those requirements to reduce the number of non-trivial rates by considering only rates with an FER between 0.05 and 0.95 as non-trivial. If the intuition behind the heuristic is correct, a small portion of rates will be non-trivial and the rest of them will be trivial and can therefore be eliminated from sampling.

In this experiment, a laptop (i.e., receiver) equipped with the wireless N USB adapter, which supports 96 transmission rates, is carried at walking speeds for 15 minutes in the office environment as described in Section 4. There exists no line-of-sight between the AP and client for most of the experiment, because the signal is blocked by obstacles such as metal cabinets, cubicle partitions and walls. We use a 2.4 GHz channel, which is exposed to WiFi and non-WiFi interference. The distance between the AP and client ranges from 1 meter to about 20 meters. In this experiment, all 96 transmission rates are sampled using a round robin ordering without frame aggregation. Frame aggregation is not used in this experiment to increase the number of samples collected for each transmission rate.

Fig. 12 shows the number of non-trivial rates that *cannot be eliminated* by the heuristic over the 15 minute experiment. The figure indicates that there are several times when the number of non-trivial rates is more than 70. Therefore, at those times the heuristic can eliminate less than 26 of the 96 transmission rates.

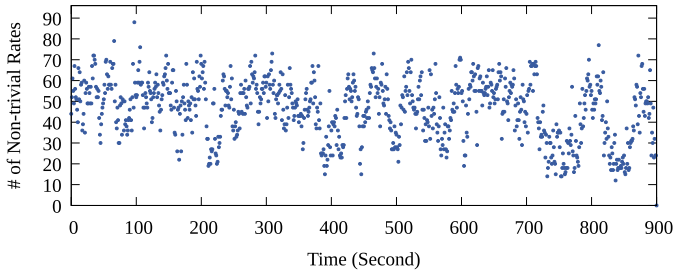


Fig. 12. Mobile 2.4 GHz.

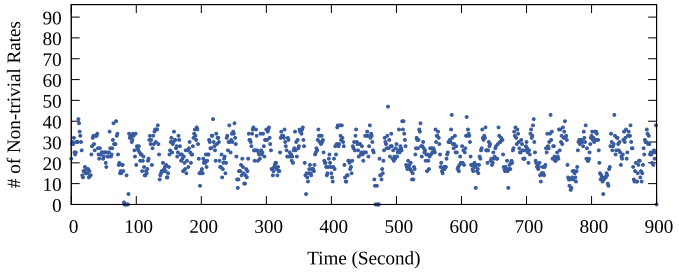
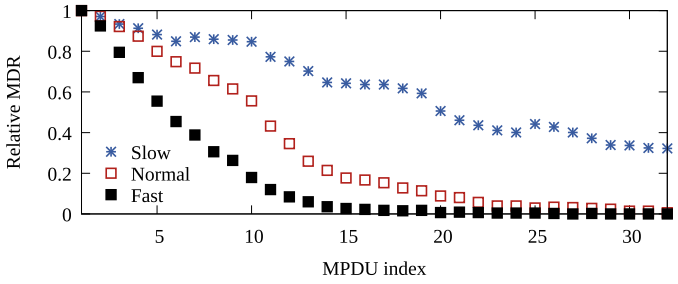
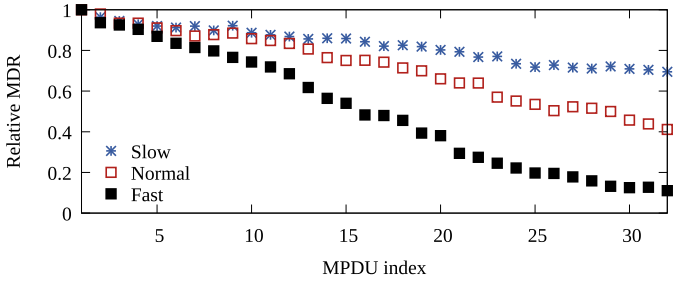


Fig. 13. Mobile 5 GHz.



(a) 5 GHz



(b) 2.4 GHz

Fig. 14. Impact of speed on MPDU delivery ratio.

Sampling 70 transmission rates using frame aggregation takes over 200 ms to complete. As a result, each rate has at most one sample over the coherence windows (i.e., 200 ms) and this heuristic, based on dynamic rate elimination, will also suffer from a lack of samples. Therefore, we dismiss this as a potential trace collection methodology. We believe that the reason for these counterintuitive results is WiFi and non-WiFi interference. We believe that interference might be strong enough to change the FER of most transmission rates regardless of the amount of redundancy encoded in the frames.

To support this claim, we repeat the experiment in the 5 GHz spectrum on a channel with no WiFi or non-WiFi interference and present the results in Fig. 13. Note that except the spectrum band, everything else in this experiment is unchanged. Fig. 13 indicates that when the interference is eliminated from the experiment, the number of non-trivial rates decreases significantly.

In summary, the rate elimination heuristic might be able to skip many transmission rates when there is no interference. However, our goal is to collect traces under a variety of channel conditions including those that are affect by different sources of interference. As a result, this heuristic cannot be used for our purpose. In the next section, we characterize the MPDU delivery ratio patterns in an A-MPDU and then use the characterization to design a methodology for collecting traces with many transmission rates.

8. Inferred measurement technique

The Frame Error Rate (FER) is not identical for the MPDUs in an aggregated frame. For this reason, the direct measurement methodology, transmits the largest possible A-MPDU, during trace collection, to measure the FER of all MPDU indexes. However, using frame aggregation during trace collection leads to the lack of samples problem in 802.11n networks with many transmission rates as explained in Section 7. We now describe our new inferred measurement methodology, which avoids frame aggregation for trace collection to increase the number of samples collected over the channel coherence window. Note that when frame aggregation is disabled, only the error rate of the first MPDU can be measured. Since during simulation an A-MPDU with any arbitrary length might be transmitted, the inferred measurement methodology needs to accurately estimate the error rate of the missing MPDUs. To explain the design of this methodology, we need to understand the characteristics of the MPDU delivery ratio (MDR) in an aggregated frame. Therefore, we first study the effect of carrier frequency and the speed of movement on the changes to the MDR in an A-MPDU, then, we describe the design of the inferred measurement technique.

8.1. Changes of MDR in an A-MPDU

Byeon et al. [7] report that if channel conditions change rapidly (for instance when the sender or receiver is moving), subframes at the end of an A-MPDU experience higher error rates than those closer to the header of the frame. In addition, they show that higher speeds of movement intensify this phenomenon. In order to accurately estimate the MDR of subframes in an A-MPDU, we need to characterize and understand how the MDR of different subframes change within an A-MPDU. *In this section, the MDR of subframe i is denoted by* MDR_i.

We design a mobile experiment where a laptop with the wireless N USB adapter is carried at different speeds from a very slow to a fast walking speed in the office environment described in Section 4. We divide the experiment into three parts, namely, slow, normal, and fast walking speeds.

Fig. 14 shows the relative MPDU delivery ratio (relative MDR) for the physical rate of 117 Mbps for the three walking speeds. We present the relative MDR for an easier comparison of the 2.4 and 5 GHz experiments. The relative MDR is computed by dividing the MDR of each index by the MDR of the first MPDU. For instance, the relative MDR of subframe i is $MDRi/MDR_1$. We can observe that in both spectrums as the speed increases the delivery ratio drops more sharply due to more rapid changes in the channel conditions, this is because the initial calibrations done on the preamble are less representative of the channel state as the time since the preamble increases. The results show that the speed has a significant role in the decrease of the MDR and it is more prominent in the 5 GHz spectrum. For instance, let's consider MPDUs with index 16, which are in the middle of the aggregated frames. In the 2.4 GHz spectrum, the relative MDR at the slow and fast speeds are 0.84 and 0.48, respectively. The relative MDR had dropped by

43%, while in the 5 GHz band, the relative MDR is 0.64 and 0.02, which is a 97% drop. Byeon et al. [7] also observed that the speed of movement intensifies the MDR decline phenomenon. However, their measurements are conducted in the 5 GHz spectrum only. Our results show that the 2.4 and 5 GHz frequency bands impact this phenomenon differently.

One reason for the faster decrease of the MPDU delivery ratio in the 5 GHz spectrum is explained by the Doppler shift. The following formula shows how the Doppler shift is directly related to the transmission frequency (f_0):

$$\Delta f = \left(\frac{\Delta v}{c}\right) f_0 \tag{2}$$

where Δf is the frequency shift, Δv is the relative speed of the sender and receiver, and c is the speed of light. As Eq. (2) indicates, for a given relative speed Δv, the frequency shift in the 5 GHz spectrum is almost twice that of the 2.4 GHz spectrum. The frequency shift due to the Doppler effect causes inter-subcarrier interference in OFDM communication systems, which consequently increases the bit error rate leading to a higher FER. As a result, in the 5 GHz spectrum, the later MPDUs in an aggregated frame are more likely to observe a change in the channel conditions (i.e., frequency shift) due to a stronger Doppler effect.

We observe that the MDR decline patterns are related to the speed of movement. If we can quantify the channel dynamics caused by the movement of the sender, receiver, or nearby objects, we might be able to estimate the MDR of all other MPDUs from the MDR of the first MPDU and the channel dynamics metric.

The inferred measurement methodology (explained later in this section) utilizes this idea by turning off frame aggregation during trace collection (i.e., measuring the MDR of the first MPDU only) and estimating the MDR of *missing* MPDUs. By not using frame aggregation, the inferred measurement methodology can collect more samples during the same time interval than the direct measurement technique. We show that the inferred measurement methodology is able to support 802.11n networks with many transmission rates (in this case up to 96). We next describe a novel technique for quantifying the channel dynamics, then we present the design of the inferred measurement methodology.

8.2. Channel dynamics indicator (CDI)

We have shown that the decline of the MPDU delivery ratio in an aggregated frame depends on the speed of movement. We now propose measuring the channel dynamics (which changes with the speed of movement) by using changes in the RSSI of ACKs. Note that we do not use RSSI to estimate the frame error rate, instead, we infer the channel dynamics from the changes in the measured RSSI.

We require a statistical measure that reflects the channel dynamics. In other words, this metric should change with the speed or amount of movement of the sender, receiver, and the surrounding objects. We will use this measure to estimate how the MDR declines as the subframe index increases (i.e., how later frames within the A-MPDU have higher error rates). We found that the variance of the difference between consecutive RSSI measurements, which we call *Channel Dynamics Indicator (CDI)*, is a suitable measure. Studying other statistical measures will be the subject of future work.

Fig. 15 shows the channel dynamics indicator for the experiment we described in Section 8.1. In this experiment, the speed of movement increases gradually during the experiment, therefore, we expect that the channel dynamics indicator to change in conjunction with the speed of movement. The graph shows that our metric correctly increases with the movement speed. At the beginning of the experiment, when the movement speed is very low, the CDI is fairly close to zero. As the speed increases, the CDI also increases to reflect the more variable channel conditions. In the next section, we show how this metric can be utilized to accurately estimate the MPDU delivery ratios.

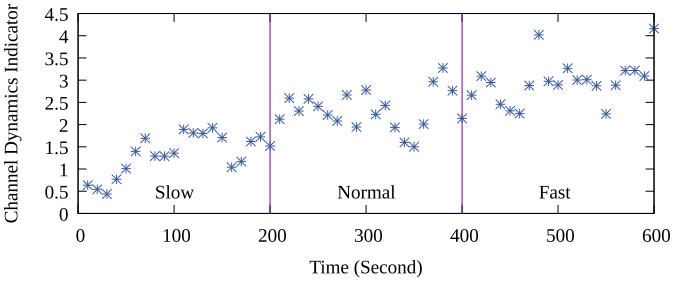


Fig. 15. Channel dynamics indicator.

8.3. Inferred measurement methodology

We observed that the channel dynamics indicator (CDI) can potentially be a good estimator for the fluctuations in the channel conditions caused by mobility. We have also shown that the MPDU delivery ratio decline in an A-MPDU depends on the speed of movement. As a result, we hypothesize that if the delivery ratio of the first MPDU of a given A-MPDU and the CDI can be determined, the delivery ratio of other MPDUs in that A-MPDU could be estimated. If this hypothesis turns out to be true, then we can collect traces without frame aggregation and infer the delivery ratio of missing MPDUs from non-aggregated frames (which correspond to the first MPDU in an A-MPDU) and CDI. In other words, we can increase the number of samples by not using frame aggregation, but still estimate the delivery ratio of all MPDUs, emulating the case where traces are collected using frame aggregation.

We now show that for the traces studied, the channel dynamics indicator and the delivery ratio of the first MPDU (MDR_1) can accurately estimate the delivery ratio of other MPDUs (MDR_i), where i is the MPDU index. In general, we observe that *for each transmission rate*, there exists a function f_i that maps the CDI and MDR_1 to MDR_i:

$$\forall i \in \{1, 2, \ldots, MAX\} \quad \exists f_i : (MDR_1, CDI) \rightarrow MDR_i \tag{3}$$

where MAX is the maximum number of MPDUs that can be transmitted in 4 ms[2]. Note that the ath9k driver further limits the number of MPDUs to 32.

Fig. 16 illustrates how the channel dynamics indicator (CDI) and the MDR of the first MPDU (MDR_1) is related to MDR_{16} (bottom) and MDR_{32} (top), for the experiment described in Section 8.1. In this experiment, a laptop is carried at various speeds for 15 minutes in the office environment.

The results for the physical layer transmission rate of 117 Mbps is presented in the graph. Similar results were observed for other transmission rates. As depicted in the figure, when the channel dynamics indicator increases (i.e., corresponding to faster movement) the MPDU delivery ratios of MPDUs 16 and 32 decrease for any given MDR_1. This is in line with our findings in Section 8.1 where we observed that the MDR drops more rapidly as the speed increases. MDR_{16} and MDR_{32} are relatively high when the receiver moves very slowly (i.e., CDI $<$ 1) and MDR_1 is high (e.g., due to good channel conditions for the given modulation and coding scheme). As expected, MDR_{16} is generally higher than MDR_{32} as MPDUs closer to the frame header have a higher delivery ratio.

[2] The ath9k driver limits the transmission time to 4 ms to comply with the channel occupancy restriction at the 5 GHz ISM spectrum.

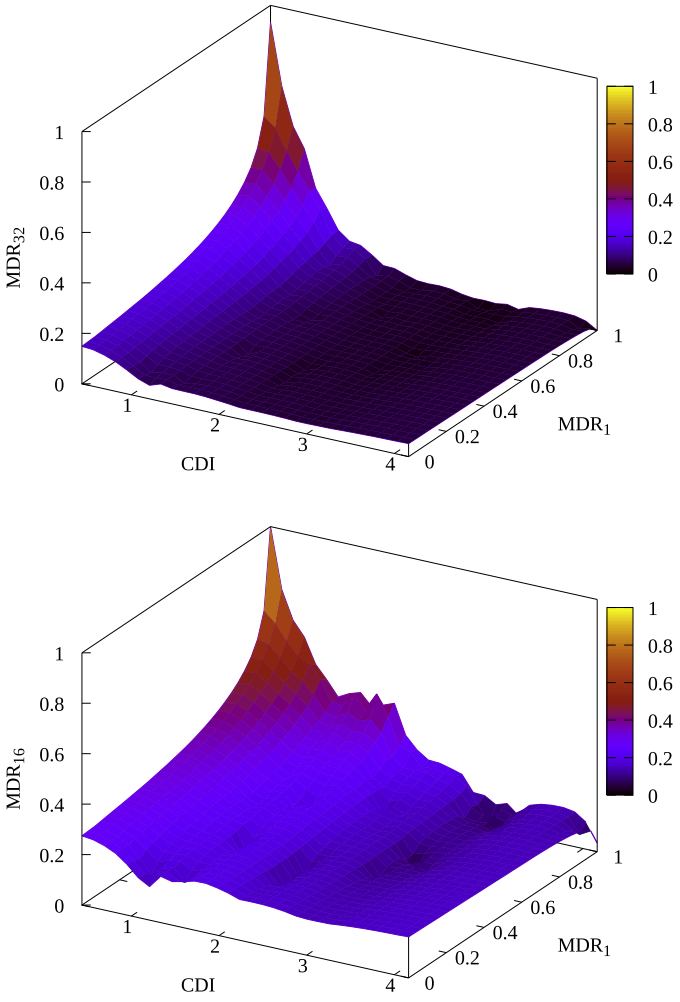


Fig. 16. Channel dynamics indicator.

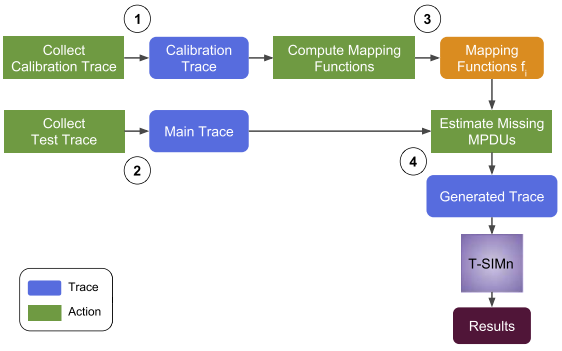


Fig. 17. Inferred measurement methodology.

8.3.1. Methodology overview

In order to enable trace collection using devices that support many transmission rates we have designed and implemented the inferred measurement methodology. The overview of our methodology (illustrated in Fig. 17) is as follows:

1. A *calibration trace* is collected using frame aggregation, using the direct measurement technique explained in Section 3.1.
   This trace is used to find the relationships between the CDI and MDRs. To compute the MDRs, we use 10-second windows to obtain enough samples to accurately measure the MDRs (i.e., up to 30 samples inside the averaging window). Note that the goal is not to measure the MDRs at a particular time but rather to compute the relationships between the CDI and MDRs. Therefore, in this case, the averaging window does not need to be smaller than or equal to the channel coherence time.
2. A *main trace* is collected without using frame aggregation (i.e., $FA_{COL}=1$).
   This trace is called the main trace because it is a trace of the experiment and environment that will be later used in the simulation (i.e., it is the experiment that will be simulated). The MDR_1 and CDI information from this trace along with the mapping functions (obtained from the calibration trace) are used in the next step to find the fate of missing MPDUs.
3. For each transmission rate and subframe index i, the calibration trace gives us a set of 3-tuples, MDR_1, MDR_i, and CDI, which creates a 3D surface (i.e., *mapping function*) as previously shown in Fig. 16. We use multi-variable regression, where MDR_1 and CDI are the estimator variables and MDR_i is the estimated variable, to determine this mapping function (details are presented later in this section). This procedure is repeated for all transmission rates and subframe indexes.
4. In order to estimate the fate of missing MPDUs in the main trace, we use the following procedure: for a given frame in the main trace at time t and transmission rate R, we consider a time window centered at time t and compute the average MDR of MPDU 1 (i.e., MDR_1) from the frames with rate R in this window. The time window we use in this study is 200 ms to roughly approximate the channel coherence time. Similarly, we compute the CDI from the RSSI of all ACKs, regardless of the transmission rate, in this window. Note that the RSSI is a property of an electromagnetic signal not the transmission rate. Then, we use the computed MDR_1 and CDI as input into the mapping functions to estimate the expected fate of the MPDUs missing from the main trace. This procedure is repeated for all frames in the main trace and the results are stored in the *generated trace*. The generated trace is similar to the output of the direct measurement methodology, except that in this case the MPDU fates are estimated (or *inferred* by applying the models obtained from the calibration trace to the frames obtained in the main trace). This generated trace is used by T-SIMn to conduct performance evaluations.

Note that both calibration and main traces include the RSSI of ACKs needed to compute the CDI. In the inferred measurement technique, the calibration trace is only used to find the mapping functions. These mapping functions are used to estimate the fate of missing MPDUs in the main trace and can be used with different main traces in that environment, therefore, they do not need to be recalculated for each new main trace. Then, the main trace is actually used for performance evaluation. The advantage of this methodology is that the main trace contains many more frames per unit of time than can be collected using the direct measurement methodology. For example, in the inferred measurement methodology, if all 96 transmission rates (in a 3x3 MIMO system) are sampled using a round robin sampling technique, we obtain one sample every 43 ms for each transmission rate (i.e., about 5 samples over a 200 ms coherence window). This is a large gain in terms of the number of samples collected when compared with trace collection using frame aggregation (i.e., direct measurement) where we sample each rate every 300 ms (i.e., less than one sample over a 200 ms coherence window). As a result, for devices that support 3 spatial streams (i.e., 96 transmission rates),the inferred measurement methodology increases the number of collected samples to 5 samples over the 200 ms coherence window, compared with only 0.7 samples that would be obtained if we were to use the direct measurement methodology. While 5 samples may seem low, we show that the inferred measurement methodology can accurately estimate the fate of missing MPDUs.

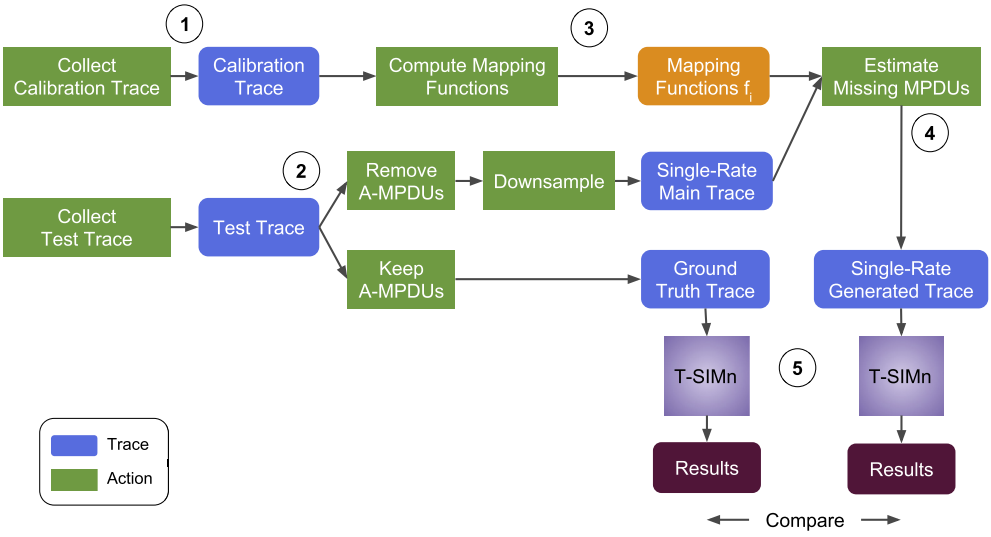


Fig. 18. Inferred measurement methodology evaluation procedure.

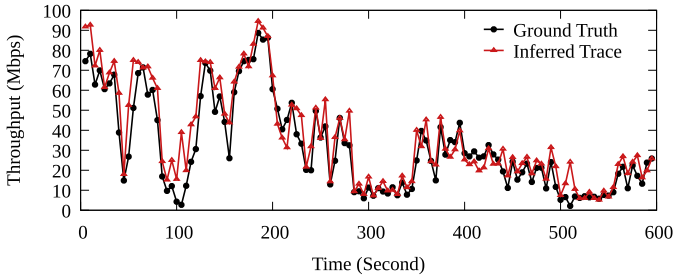
Eq. (4) shows the regression model used in this study to obtain the mapping functions from the calibration trace. The model that we use contains two predictor terms MDR_1 and CDI. In addition, it has quadratic and interaction terms. We now explain why we include these terms in the model. Fig. 16 shows that the relation between the predictor terms (i.e., MDR_1 and CDI) is curvilinear with MDR_i. As a result, we add the quadratic terms to the regression model. For the traces used in this study, we did not find higher order terms necessary in the regression model. In addition, as can be seen in Fig. 16, there are interactions between the predictor terms. For example, when $CDI = 4$, the relation between MDR_1 and MDR_{32} is almost constant (i.e., $MDR_{32} \approx 0$). On the other hand, when $CDI = 0$, MDR_{32} is a curvilinear function of MDR_1. This observation confirms the existence of interactions between the predictors. As a result, we add the $MDR_1{*}CDI$ term to Eq. (4) to model the interactions between MDR_1 and CDI. We will show that the regression model in Eq. (4) works well for the purpose of this study, however, we plan to examine other possible models in future work.

$$MDR_i = \beta_0 + \beta_1 MDR_1 + \beta_2 CDI + \beta_3 {MDR_1}^2 + \beta_4 CDI^2 + \beta_5 (MDR_1 * CDI) \quad (4)$$

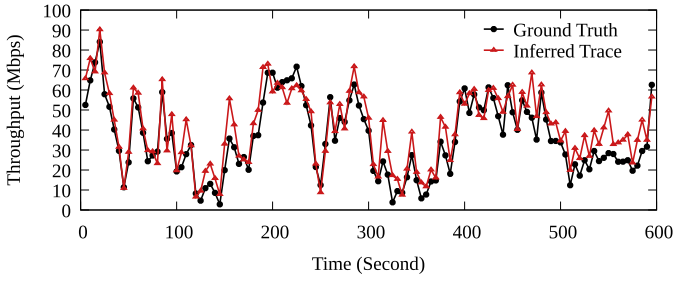
8.4. Methodology evaluation

To evaluate the efficacy of the inferred measurement methodology one can compare the outcome of this technique with that of an actual experiment (i.e., the ground truth). However, it requires repeatability between the trace collection and the ground truth experiment and achieving repeatability is very difficult, especially in environments that involve mobility and interference. As noted previously, our goal is to be able to simulate environments with mobility and interference. Therefore, we design a novel technique, illustrated in Fig. 18, to study the accuracy of the inferred measurement methodology. Instead of using the main trace, which contains samples for all rates, we collect a trace (called a test trace) for a given transmission rate by alternatively sending frames with and without frame aggregation. The frames without frame aggregation serve as the input to inferred measurement methodology, while the aggregated frames are used to determine ground truth. The frames without frame aggregation are down-sampled before being fed to the inferred measurement methodology to match the number of samples obtained normally when sampling all transmission rates (i.e., matching the number of usable frames that would be available in the main trace). With this approach, we run an experiment in a way that it gives us the ground truth data and the input to the inferred measurement methodology simultaneously. The inferred measurement methodology should be able to use only those non-aggregated frames to accurately estimate the fate of each MPDU in aggregated frames. Remember that with the inferred measurement methodology, in order to collect the main trace, we disable frame aggregation to increase the number of samples obtained per unit time. Then, we infer the fate of aggregated frames using the mapping functions (obtained from the calibration trace) and the main trace (which contains non-aggregated frames only). We now explain the procedure for evaluating the inferred measurement methodology step by step as illustrated in Fig. 18. Note that some steps are identical to what is done in the inferred measurement methodology (marked as [IDENTICAL]).

1. Collect a calibration trace as explained before (i.e., round robin all transmission rates with $\texttt{FA}_{\texttt{COL}}$=MAX). [IDENTICAL]
2. Collect a test trace for a given transmission rate R. Frame aggregation is turned on and off alternatively (i.e., $\texttt{FA}_{\texttt{COL}}$=MAX and $\texttt{FA}_{\texttt{COL}}$=1) for each physical frame. Aggregated frames are extracted from the test trace and are stored in the *ground truth trace*, while non-aggregated frames are stored in the *single-rate main trace*. As opposed to the main trace, which is normally collected when using the inferred measurement methodology (containing the fate of all rates), this single-rate main trace contains the fate of rate R only. Note that for the purpose of evaluating the inferred measurement methodology, instead of sampling all rates, we sample only one rate so that we can also collect the ground truth data (i.e., aggregated frames) simultaneously.
3. For each transmission rate, mapping functions f_i are computed using the regression model in Eq. (4) based on the calibration trace MDRs and CDI data [IDENTICAL].
4. To estimate the MDR of missing MPDUs, the MDR_1 and CDI from the single-rate main trace are used as input into the mapping functions, which are used to produce the *single-rate generated trace*.
5. The single-rate generated trace and ground truth traces are fed separately to T-SIMn. T-SIMn is configured to saturate the link with a UDP stream using the constant physical layer rate R.



(a) 5 GHz



(b) 2.4 GHz

Fig. 19. Inference measurement methodology evaluation.

We compare the throughput obtained from the generated and ground truth traces. If the inferred measurement technique is able to accurately estimate the MDR for A-MPDUs, then the obtained throughputs should match.

To evaluate the inferred measurement methodology using this procedure, we use a laptop with the wireless N USB adapter (which supports 96 transmission rates) as the mobile receiver in the office environment as described in Section 4. The laptop (i.e., receiver) is carried from a very slow to a very fast walking speed for 10 minutes to collect the calibration trace and once more to collect the test trace. The traces are collected at the sender, which is the stationary access point. In this experiment, there is a 2 week gap between the collection of the calibration and test traces to test the robustness of the calibration trace.

Fig. 19 shows the throughput obtained from T-SIMn for the generated trace, produced by the inferred measurement methodology and the ground truth trace for experiments conducted using 2.4 and 5 GHz spectrums. Every point on the plots shows the average throughput computed over a 5 second window. The figures show that despite the highly dynamic environments, the throughput obtained from the inference methodology closely matches the ground truth data.

Software retransmission is disabled in T-SIMn in this experiment so that the block ACK window advancement does not limit the length of the transmitted A-MPDUs. As a result, the simulator always transmits full-length A-MPDUs to ensure that the estimated fates of all 32 MPDUs are used in the evaluation. Despite the 2 week time gap between the collection of the calibration and test traces and possible environmental changes, the inferred measurement methodology produces accurate results. We speculate that if the calibration and main traces are collected more closely in time, more accurate results may be possible. We recommend that calibration traces include the same or bigger range of speeds of movement as the main trace so that a better fit can be achieved when performing the regression to compute the mapping functions. In future work, we intend to study if calibration is required for different environments.

Note that a particular transmission rate (i.e., 117 Mbps) is used in the *evaluation of the inferred measurement methodology*. This rate was chosen because we believe it would be a challenging choice. To see if this is in fact a challenging choice, we consider the number of *non-trivial rates* (as introduced in Section 7.1). If a rate is *trivial* (i.e., the FER is almost 0 or 1) for most of the time during an experiment estimating missing MPDUs for this rate is a fairly easy task as the FER of missing MPDUs is also most likely 0 or 1. On the other hand, estimating the fate of missing MPDUs for a *non-trivial* rate is much more challenging.

We define Non-trivial FER ratio as the ratio of the FER measurements for a transmission rate with the FER between 0.05 and 0.95 to the total number of FER measurements for that rate during an experiment. We consider a rate to be non-trivial at a given time, if the FER of this rate is between 0.05 and 0.95 over a one second window centered at that time. To quantify how challenging transmission rates are for the purpose of evaluating our new methodology, we compute the ratio of time each rate is non-trivial. For instance, if the FER of a rate is between 0.05 and 0.95, for 40% of the measurements, the *non-trivial FER ratio* for this rate is 0.4 in that particular experiment. We compute the non-trivial FER ratio for all 96 rates in the calibration traces used in the 2.4 and 5 GHz experiments. In the 2.4 GHz trace, the rate we used in the evaluation (i.e., 117 Mbps) has the highest non-trivial FER ratio (i.e., 0.72). In the 5 GHz trace, this rate has a ratio of 0.81 and is the eighth highest non-trivial FER ratio (the highest ratio is 0.89). The high non-trivial FER ratios for the 117 Mbps rate make this rate a good candidate for the evaluation of our methodology, because estimating the missing MPDUs is challenging for this rate.

When collecting the test traces, we continuously monitor the FER to keep it in the non-trivial range. For instance, when the FER approaches 0.95, we start moving back towards the access point to lower the FER. As a result, the non-trivial FER ratio of the 117 Mbps rate in the 2.4 and 5 GHz test traces are 0.997 and 0.998, respectively. We believe that the rate used in this evaluation is a suitable choice for the purpose of this study due to the high non-trivial FER ratios and attempts made to keep the FER in the challenging non-trivial range.

Traces for the direct measurement methodology are easy to collect and when applicable this methodology is the preferred method, however, it is applicable only to 802.11 networks with a smaller number of transmission rates. We show that the direct measurement methodology can be used to collect traces with an iPhone that supports 32 transmission rates. To support networks with many more transmission rates such as MIMO 3x3:3 systems, we have designed the inferred measurement methodology. This technique utilizes the relationship between the MDR of MPDUs within an A-MPDU and the channel dynamics indicator to enable us to collect traces with fame aggregation disabled (thus increasing the number of samples collected per unit of time), and to infer the MDR for MPDUs that would have been sent with frame aggregation. Finally, we have utilized T-SIMn to evaluate the accuracy of the inferred measurement methodology. Our trace collection methodologies enable the T-SIMn framework to accurately measure the channel conditions in a variety of environments including those with WiFi and non-WiFi interference and mobility. When comparing the performance of two or more systems or algorithms using T-SIMn, utilizing traces guarantees that the competing alternatives are exposed to exactly the same channel conditions. Therefore, any differences observed in the performance are solely due to differences in those algorithms or systems and not due to differences in channel conditions. To evaluate the ability of traces to accurately capture fast changing channel conditions, we demonstrate that the inferred measurement methodologies achieve a high degree of accuracy. It is worth noting that less accurate traces could

still be used to evaluate competing alternatives as they would be compared using the same trace.

9. Conclusions

In this paper, we design a trace-based simulation framework for 802.11n networks, called T-SIMn. We find that carefully considering all factors that affect throughput such as 802.11n transmission features, channel access, and channel error rate are 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 design and evaluate two trace collection methodologies, namely, direct measurement and inferred measurement techniques. Using the direct measurement methodology, we demonstrate that the T-SIMn framework can be used with 1 antenna devices (which includes most smart phones and tablets). Finally, we show that the inferred measurement technique enables the framework to work with devices that support many transmission rates such as MIMO 3x3:3 systems.

We have implemented the frame aggregation and Minstrel HT rate adaptation algorithms from the ath9k driver in our simulator. In addition, we have implemented the MoFA frame aggregation algorithm [7], are currently working on the STRALE rate adaptation and frame aggregation algorithms [26] and plan to add more algorithms. We plan to use T-SIMn to conduct a comprehensive evaluation of existing link adaptation and frame aggregation algorithms using a wide variety of environments. Additionally, as part of our future work, we plan to compare the accuracy of T-SIMn with that of simulators like ns-3and OMNeT++. We believe that T-SIMn is a valuable tool for the realistic and repeatable performance evaluation of 802.11n networks. To that end, we have made some traces available [27] and intend to add more traces and T-SIMn to this repository.

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.

References

[1] A. Abedi, T. Brecht, T-RATE: A framework for the trace-driven evaluation of 802.11 rate adaptation algorithms, MASCOTS, 2014.
[2] L. Kriara, M.K. Marina, SampleLite: a hybrid approach to 802.11n link adaptation, ACM SIGCOMM Comput. Commun. Rev. 45 (2) (2015) 4–13.
[3] L. Deek, E. Garcia-Villegas, E. Belding, S.-J. Lee, K. Almeroth, A practical framework for 802.11 MIMO rate adaptation, Comput. Netw. 83 (2015) 332–348.
[4] Z. Zhao, F. Zhang, S. Guo, X.-Y. Li, J. Han, RainbowRate: MIMO rate adaptation in 802.11n WiLD links, in: Proceedings of the IPCCC, 2014.
[5] L. Deek, E. Garcia-Villegas, E. Belding, S.-J. Lee, K. Almeroth, Intelligent channel bonding in 802.11n WLANs, IEEE Trans. Mob. Comput. 13 (6) (2014) 1242–1255.
[6] P. Huang, X. Yang, L. Xiao, Adaptive channel bonding in multicarrier wireless networks, in: Proceedings of the MobiHoc, 2013.
[7] S. Byeon, K. Yoon, O. Lee, S. Choi, W. Cho, S. Oh, MoFA: mobility-aware frame aggregation in Wi-Fi, in: Proceedings of the CoNEXT, 2014.
[8] P. Teymoori, A. Dadlani, K. Sohraby, K. Kim, An optimal packet aggregation scheme in delay-constrained IEEE 802.11n WLANs, in: Proceedings of the WiCOM, 2012.
[9] A. Abedi, A. Heard, T. Brecht, Conducting repeatable experiments and fair comparisons using 802.11n MIMO networks, ACM SIGOPS 49 (1) (2015) 41–50.
[10] R. Burchfield, E. Nourbakhsh, J. Dix, K. Sahu, S. Venkatesan, R. Prakash, RF in the jungle: effect of environment assumptions on wireless experiment repeatability, in: Proceedings of the ICC, 2009.
[11] M. Lacage, T.R. Henderson, Yet another network simulator, Proceeding from the 2006 Workshop on ns-2: The IP Network Simulator, ACM, 2006.
[12] A. Varga, R. Hornig, An overview of the OMNeT++ simulation environment, in: Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops, in: Simutools '08, 2008, pp. 60:1–60:10.
[13] R. Technology, SteelCentral Riverbed Modeler, 2016, http://www.riverbed.com/products/steelcentral/steelcentral-riverbed-modeler.html.
[14] Scalable Network Technologies, Inc., QualNet, 2014, http://web.scalable-networks.com/qualnet.
[15] M. Stoffers, G. Riley, Comparing the ns-3 propagation models, in: Proceedings of the MASCOTS, IEEE, 2012, pp. 61–67.
[16] INET Framework for OMNeT++, January 22, 2016, 2016https://omnetpp.org/doc/inet/api-current/inet-manual-draft.pdf.
[17] G. Judd, P. Steenkiste, Repeatable and realistic wireless experimentation through physical emulation, SIGCOMM Comput. Commun. Rev. 34 (1) (2004) 63–68.
[18] G. Judd, P. Steenkiste, Using emulation to understand and improve wireless networks and applications, in: Proceedings of the NSDI, 2005.
[19] J. Zhou, Z. Ji, R. Bagrodia, TWINE: A hybrid emulation testbed for wireless networks and applications., in: Proceedings of the INFOCOM, 2006.
[20] D. Halperin, W. Hu, A. Sheth, D. Wetherall, Predictable 802.11 packet delivery from wireless channel measurements, ACM SIGCOMM Comput. Commun. Rev. 41 (4) (2011) 159–170.
[21] D. Kotz, C. Newport, R.S. Gray, J. Liu, Y. Yuan, C. Elliott, Experimental evaluation of wireless simulation assumptions, in: Proceedings of the MSWiM, 2004.
[22] V. Lenders, M. Martonosi, Repeatable and realistic experimentation in mobile wireless networks, IEEE Trans. Mob. Comput. 8 (12) (2009) 1718–1728.
[23] W.-L. Shen, Y.-C. Tung, K.-C. Lee, K.C.-J. Lin, S. Gollakota, D. Katabi, M.-S. Chen, Rate adaptation for 802.11 multiuser MIMO networks, in: Proceedings of the Mobicom, 2012.
[24] J.G. Andrews, A. Ghosh, R. Muhamed, Fundamentals of WiMAX-Understanding Broadband Wireless Networking, Prentice Hall, Upper Saddle River, NJ, 2007.
[25] A. Heard, T-SIMn: Towards a Framework for the Trace-Based Simulation of 802.11n Networks, University of Waterloo, 2016 Master's thesis.
[26] S. Byeon, K. Yoon, C. Yang, S. Choi, STRALE: mobility-aware PHY rate and frame aggregation length adaptation in WLANs, in: Proceedings of the Infocomm, 2017.
[27] Sample traces, https://cs.uwaterloo.ca/~brecht/t-simn.