A Lightweight Scheme for Rapid and Accurate WiFi Path Characterization

Lixing Song, Aaron Striegel
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46556
Email: lsong2@nd.edu, striegel@nd.edu

***Abstract*—WiFi serves as one of the key mechanisms for wireless access for mobile devices whether at home, on travel, or during normal day-to-day activities. Unfortunately, the perceived high bandwidth and low cost of WiFi is often tempered with varying degrees of quality. Compounding this further, existing techniques for assessing network performance are often expensive in terms of time, bandwidth, and energy making them ill-suited for widespread, longitudinal deployment. In this paper, we propose Fast Mobile Network Characterization (FMNC) to address this shortcoming. FMNC uses sliced, structured, and reordered packet sequences along with an awareness of frame aggregation to rapidly characterize available bandwidth. FMNC does this within the context of a single HTTP GET, consuming less than 100 KB on the downlink with resolution of the path characteristics typically occurring in under 250 ms. We demonstrate the performance of FMNC through extensive lab experiments under a variety of configuration scenarios.**

I. Introduction

For most users, WiFi has become synonymous with high speed, low cost wireless network access. Although cellular access has gradually transitioned towards nearly unlimited plans over the past few years, cellular data speeds often pale in comparison to good WiFi. Unfortunately, good WiFi can often be quite difficult to find. Despite most cellular operators aggressively pushing users to join WiFi, the resulting quality from said WiFi may not always be better. For many users, the WiFi Quality of Experience (QoE) can often fall woefully short versus slower but more consistent cellular network access.

To that end, there exists a wide body of work on network characterization with more recent work on leveraging multi-path options spanning WiFi and cellular networks. Although a mobile device has no control over user mobility, the mobile device does have control with respect to choosing WiFi or cellular and the extent to which particular WiFi network (SSID) might be chosen. Hence, link and / or path characterization could offer some insight as to the selection of the right network to utilize. Critically, much of existing literature fails though for the specific problem of WiFi choice in the moment. Throughput tests such as *iperf* [1], *Speedtest.net* [2], and others [3] are decidedly not timely, taking on the order of seconds and consuming significant bandwidth and energy. Crowd sourcing from past history may have some value [4]–[6] but only if current network conditions are well modeled by past performance though such results tend to be highly time/event dependent. Finally, although lightweight techniques exist for characterization [7]–[9], such techniques fare quite poorly under modern WiFi approaches (802.11n, ac).

Thus, what is the user or mobile device left to do when faced with trying to resolve the quality of WiFi quickly and accurately? It is this dilemma that forms the motivation for our paper. In our paper, we propose a scheme, Fast Mobile Network Characterization (FMNC), for rapid, accurate, and lightweight path characterization to advise with regards to WiFi performance.

The foundation of FMNC is built on the notion of sliced, structured, and reordered packet sequences that effectively tease out bandwidth limitations by inducing frame aggregation on the WiFi link in tandem with the reflected packet acknowledgment rates. We leverage RIPPS [10] for the idea to slice packets across WiFi and TCP Sting [11] for packet reordering to increase the effective measurement points, all the while operating within a prototypical HTTP response inside of normal TCP functionality. The end result is that we can rapidly infer the available path bandwidth with a precise capture between 1 to 10 Mb/s with accurate classifications for low (red zone, $<$ 1 Mb/s) and high (green zone, $>$ 10 Mb/s) speeds. Hence, the contributions of our paper are as follows:

- We describe Fast Mobile Network Characterization (FMNC) and show how the scheme can aggressively and accurately resolve path available bandwidth in under 250 ms while using less than 100 KB of data. Furthermore, we accomplish this without modifications to the end client operating inside a normal HTTP web fetch.
- We show how inducing frame aggregation (introduced by 802.11e) for 802.11n and 802.11ac networks can be used to help infer available bandwidth.
- We demonstrate through lab experiments the accuracy of FMNC across a variety of network speeds and bottleneck locations (802.11g link, 802.11n link, broadband link). We also show how FMNC can accurately recognize severe uplink congestion to avoid link misclassification.

II. Related Work

The area of network performance characterization has seen extensive research ranging back over several decades of research. Harkening back to the early days of TCP design, the notion of how bandwidth is available or should be used on

978-1-5386-5156-8/18/$31.00 ©2018 IEEE

the path is a fundamental question for network researchers. For this paper, we are concerned with two categories of work: bandwidth estimation across an end-to-end path and crowdsourcing of wireless measurements. The intuition behind these two categories is that the first category provides the toolset available to the second category. FMNC straddles both areas by providing a new scheme for the first category coupled with a large-scale deployment result that falls into the second category.

Bandwidth Estimation: Foundationally, the notion of end-to-end bandwidth estimation can be reduced to the problem of trying to predict what performance can be achieved across said path [7], [12]–[15]. Hence, the simplest form is to start a flow and then to fully flex the speed until the path is characterized. *iPerf*, *Speedtest.net*, and others [3] fall into the category of Achievable Throughput (AT) measures.

Conversely, there is a large body of research that attempts to divine network performance without conducting an actual AT test. Nominally, most works in this category focus on characterizing the Available Bandwidth (AB) which represents the spare link capacity rather than the achievable throughput. We further discuss the distinction of AT vs. AB later in the paper. The PRM (Packet Racket Model) approach (ex. PathLoad [14], PathChirp [7], and TOPP [16]) send traffic with increasing rates to find the point where link congestion starts in order to approximate the available bandwidth.

In contrast, the Packet Gap Model (PGM) approach (ex. IGI [15], Spruce [8] and Abing [17]) estimate the available bandwidth based on an analysis of the dispersion between consecutive packets at a receiver. More recently, Wbest [9] proposed a novel method explicitly for WiFi with a more recent variation (WBest+ [18]) improving performance by minimizing the frame aggregation introduced by 802.11e. For FMNC, rather than avoiding frame aggregation as WBest+ attempts to do, we seek to induce frame aggregation and then measure said aggregation to help identify the available bandwidth.

Crowdsourcing: One alternative to direct observation is to defer such efforts to the crowd. Given the widespread prevalence of WiFi on mobile devices, crowdsourcing network performance would seem to be an excellent fit. In [4], the authors proposed MCNet, a tool that allows users to crowd-source Wi-Fi performance measurements. The authors in [19] proposed a similar study at the city scale and revealed several problems in WiFi deployments in public spaces. Furthermore, in the context of MPTCP, the works of [6] and [20] explored characterizing WiFi versus cellular networks. To the best of our knowledge, our paper is one of the first works to present a large-scale crowd-sourced deployment involving available bandwidth. We intend following the publication of the paper to anonymize and release all data via CRAWDAD which includes both our core FMNC AB test as well as ground truth AT observations.

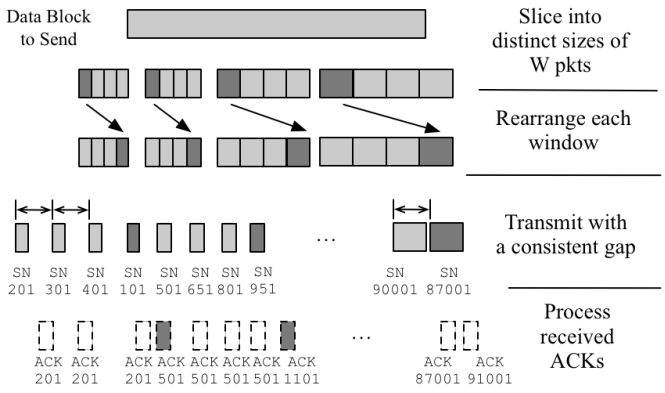

Fig. 1. Sliced, Structured, and Reordered Packets

III. FMNC Overview

At a high level, the goal of FMNC is to quickly and accurately determine the performance of a path with WiFi that is available to a mobile device. Whether that link is being evaluated as the exclusive link (ex. joining a public WiFi), being used in a multipath manner (ex. MP-TCP), or is being periodically monitoring, FMNC should provide significant insight as to the expected performance of a given path inclusive of WiFi. We define the following design principles for FMNC:

- *Work within existing confines:* No changes should be required to the client or network infrastructure. All requests or operations should operate within existing networking protocols.
- *Focus on speed ranges that matter:* Accurate path characterization is essential when speeds are slow (< 10 Mb/s) but less essential when speeds high (> 10 Mb/s). Bottlenecks may occur at any point on the path leading to or coming from the WiFi serving the mobile device.
- *Resolve quickly:* Path characterization must happen on the order of hundreds of milliseconds, not tens of seconds in order to be actionable.

A. FMNC Architecture

We first begin by describing architecturally how FMNC is set up and how the characterization techniques are broadly structured. To start, the FMNC process begins when a client initiates a TCP connection to make an HTTP request to the FMNC server[1]. The normal TCP handshake occurs followed by an HTTP GET message which contains settings for FMNC embedded as parameters in the request. The *libpcap*-based FMNC server then constructs a sliced, structured, and re-ordered packet sequence (train) for conducting an Available Bandwidth (AB) path characterization to the client.

Figure 1 presents an overview of the packet train as constructed. In conjunction with the figure, we expand on what we mean by the terms, sliced, structured, and reordered. Critically, such an approach is necessary as a traditional

[1]The usage of a non-standard port may be desirable to avoid proxying effects

pair-wise observation from a TCP flow would result in an insufficient number of observations. Typical TCP behavior will produce a ratio of data to ACK packets on the order of 2:1 or beyond. To augment this number, we apply two techniques, the slicing from RIPPS [10] and the reordering from TCP Sting [11]. First, packets are sliced into smaller sizes with the packet size being set to achieve a particular rate for a fixed size window of packets (see later). Second, packets within a given window are then reordered whereby the first packet is shuffled to the end of the window. This effect as originally identified by TCP Sting leverages TCP Fast Retransmit to yield a 1:1 ratio of data to ACK packets. In short, TCP Fast Retransmit engages if the same ACK number is seen three times implying a missing packet. By shuffling the packet to the end of the window, the first packet appears to be missing (lost) and hence we tease out additional ACKs thereby dramatically increasing our observation opportunities. As we control the server via *libpcap*, we are free to ignore the spurious ACKs.

The structured nature of the packets then comes from how the packet train is transmitted to the client. Rather than waiting as with a normal TCP sequence before transmitting packets, we immediately transmit the packet train after receipt of the HTTP GET. Packets are transmitted at a fixed interval with the packet gap P_G denoting the gap from when a packet is transmitted to when the transmission of the next packet should be started at the server. Within a given window of W packets, the packet size can then be varied to target a specific rate. For the purpose of this paper, we use five rate windows where our rate windows are set to 1, 3.25, 5.5, 7.75, and 10 Mb/s with a window size of 20 to cover the desired range of bandwidth characterization ($L_{AB} = 100$ total packets). The packet train is transmitted quickly to the client whom then responds with individual TCP ACKs for each downstream data packet. The distilling of the available bandwidth from said ACK stream forms the core novelty of our paper and is described in the subsequent sections of the paper.

B. Link Characterization - AB vs. AT

For the purposes of FMNC, the AT approach is far too heavyweight imposing a heavy burden with respect to time, bandwidth, and energy. Furthermore, while such tests are helpful for the client conducting the characterization, there considerable ancillary effects imposed on other mobile devices using the same WiFi network. In contrast, the *Available Bandwidth* of a link is a metric that describes the *spare* or *residual* capacity of the link during a certain period of time. For an end-to-end path, AB refers to the available bandwidth of the narrow link[2] which has the minimum available bandwidth. Given a L-hop end-to-end path, assume C_i is link capacity of i-th link, and $\overline{u_i}(t-\tau, t)$ is the average utilization of the link from time $t-\tau$ to t. Therefore, we define the instantaneous available bandwidth AB^t at time t for the path as:

$$AB^t = \min_{i=1,\ldots,L} C_i(1 - \overline{u_i}(t-\tau, t)) \quad (1)$$

[2]It should be noted that *narrow* link is talking to available bandwidth, which is different from a *tight link* where minimal capacity occurs.

Notably, while AB techniques can be exceptionally lightweight, careful design of the AB mechanism must consider key queuing properties of the underlying network path. For WiFi, as we will demonstrate shortly in the paper, frame aggregation imposes a deleterious effect on to the accuracy of existing AB techniques. However, before we further discuss frame aggregation, it is important to correctly distinguish what AB captures versus what AB does not capture, particularly as it relates to AT.

C. Elasticity

Effectively, AB represents the minimum throughput that a TCP flow could achieve at that point in time while AT represents what it would have actually achieved at that point in time. We dub the difference between the two terms as the *elasticity* of the link drawing from the fact that the flexibility of a given link between AB and AT will be a function of the existing flows (number, relative RTT, relative link qualities). Although that information would be nearly impossible for a mobile client to know, the concept of elasticity can be helpful to illustrate the difference between AB and AT by categorizing the key zones or cases where such differences occur:

- $AB = AT$: AB will equal AT when either the link is entirely open (no existing traffic) or all existing traffic on the link is UDP-based (not TCP friendly). A newly formed TCP flow would not be able to easily crowd out the existing traffic and thus residual capacity represents actual capacity. In practice, such a case would be quite rare.
- $AB < AT$: The most common case will be where AB represents the minimum bandwidth achievable at that point in time.
- $AB > AT$: For cases where rate-limiting may be in place, the lightweight nature of an AB test may not trip the rate-limiting features of a link. Hence, AB may actually exceed AT.

Finally, it is important to note before we continue with our mechanism for computing AB that all network characterizations are accurate for only that specific point in time and will *always* represent a lagging indicator. When coupled with the known dynamics of wireless and in particular WiFi, there will be always cases where we will get the answer wrong. However, we believe there are two important mitigating factors for our work that make our approach intriguing. First, our solution is extremely fast allowing one to run one or more characterizations in a short period of time. Second, we emphasize caution of optimism (part of the rationale for AB) giving the server and mobile client the equivalent of a *sniff* test as to quality.

D. Why WiFi and AB Mix Poorly

For many of the existing AB techniques, available bandwidth is inferred through the observation of timing changes amongst various packet trains. Techniques such as PathChirp

[7], Spruce [8], and others [9], [15], [17] leverage various characteristics of said packets to infer link characteristics. Notably, more recent WiFi protocols such as those that utilize *frame aggregation* as introduced in 802.11e (802.11n, 802.11ac) play havoc with the received timing characteristics. In short, frame aggregation bundles together multiple frames with the same destination address (MAC address) in order to reduce the control overhead (PHY-layer ACKs) and to reduce the DCF competition for the benefit of increased throughput. Ancillary benefits also include improved energy efficiency.

While such an effect can significantly improve performance over the last WiFi hop, the results are catastrophic for many existing AB techniques as the timing variations that formed the foundation for inference are now eliminated. As will be shown later in our lab experiments and has also shown via WBest+ [18], nearly all techniques that are unaware of said frame aggregation provide wildly inaccurate results as the links look dramatically more capable by virtue of the reduced inter-packet times. While WBest+ attempts to eliminate the presence of frame aggregation during measurement, FMNC actively embraces frame aggregation.

E. Embracing Frame Aggregation for AB

Consider then the mechanics of how frame aggregation ends up being applied to a sequence of packets. To start, consider the unloaded link case and the degree to which frame aggregation would exert an influence. Rather than demonstrating a work conserving behavior as most AB techniques would expect (if there is data to transmit, transmit any waiting data), APs with 802.11e exhibit non-work conserving properties in unloaded link cases for the purpose of introducing frame aggregation.

Upon receipt of a packet P from an upstream link bound for a WiFi client, a brief timer (hundreds of microseconds) is set to allow another packet to arrive for the same client before the transmission which we term T_{FAD} (Frame Aggregation Delay). T_{FAD} takes advantage of the natural behavior of TCP to bundle small bursts of packets without requiring a significant delay allowing for a wait on the order of microseconds rather than milliseconds. For example, packets sent during TCP slow start will be transmitted back-to-back by virtue of the single ACK yielding two or more back-to-back data packets. Similarly, downstream packets during congestion avoidance tend to also be back-to-back as a single ACK frequently acknowledges multiple packets resulting in a burst downstream data transmissions from the server.

Conversely, when the link is reasonably well loaded ($AB < AT$), natural queuing effects will dominate rather than T_{FAD} as cross-traffic will block between successive frame aggregations (via the MPDU) to a client but may not necessarily block between successive packets to a client. Hence, we posit that the presence of aggregation or lack thereof can be used as an indication of congestion in regions of interest. However, before we explore the usage of aggregation, we begin with the mechanism to quantify said aggregation and its robustness with respect to WiFi mechanism as well as our ability to distinguish the presence (or lack thereof) with regards to cross-traffic.

We define a metric, *Aggregation Index* (AI), to measure the degree of frame aggregation present within a given window of packets. AI measures frame aggregation over a window of N contiguous packets with T_{FAD} representing the maximum temporal spacing at which the uncongested wireless link would exhibit minimal though non-zero aggregation[3]. For this metric, let g_i be the gap between the ith and $i+1$-th packet. Then, let AI_A be the set of all packet gaps where $g_i < T_{FAD}$. The computation of AI is:

$$AI = \frac{|AI_A|}{|N-1|} \tag{2}$$

As AI trends towards 1, all packets have been aggregated while as AI approaches zero, no packets have been aggregated. Normal packet variations will push AI above zero in most cases while limitations on the MPDU limit AI from truly approaching 1.

To that end, we conduct a simple experiment to illustrate T_{FAD} shown in Figure 2. For this experiment, we construct a simple WiFi network as used in the later lab experiments. A single server sends a TCP flow utilizing the FMNC approach (structured, sliced, reordered packets) across an 802.11n network using a channel in 2.4 GHz at relatively uncongested hours with varying time intervals (300 microseconds to 2000 microseconds). Aggregation events were confirmed through packet capture utilizing an Airpcap adapter on a separate notebook. The figure shows the Empirical PDF of all inter-arrival times for ACKs as observed by the server. While aggregation may still occur due to variations in the underlying wireless medium at larger packet intervals, the dominant aggregation occurs when packets are spaced exceptionally close together. For the purposes of our paper, we use a T_{FAD} of 400 microseconds gleaned through significant in-lab experiments across a variety of APs (Cisco, Aruba, NetGear, etc.).

With $T_{FAD} = 400$ microseconds, consider a similar experiment comparing 802.11n and 802.11g as shown in Figure 3. For each data point, 20 runs were collected with each run containing 100 packets. Minimum sized packet payloads are sent with the spacing between the respective packets (spacing defined as the separation between the start of each packet in terms of passing to the network adapter at the server). In the figure, the packet spacing is varied from 300 microseconds (0.3 ms) to 2000 microseconds (2.0 ms). Notably, while the 802.11g level of AI followed a distinctively linear decrease across the various settings (AI representing small wireless 'burps), the 802.11n setup exhibited a much sharper characteristic with respect to aggregation. At roughly 1.0 ms, much of the link dynamics are removed leaving only wireless link dynamics to occasionally incur frame aggregation. We observe that a minimum packet spacing of 1.0 ms is strictly needed to avoid unnecessary frame aggregation.

[3]Wireless and upstream network dynamics make the exclusion of all aggregation effectively impossible

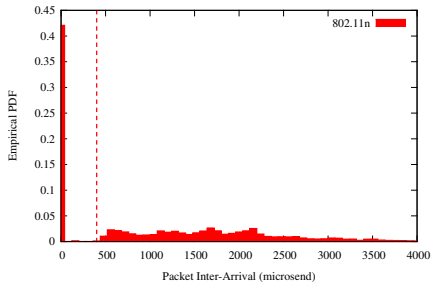

Fig. 2. Empirical PDF of Inter-ACK Delay (802.11n, 2.4 GHz)

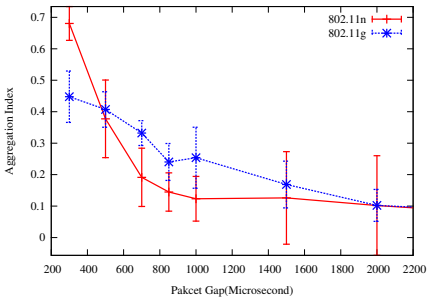

Fig. 3. Aggregation Index vs. Packet Gap (2.4 GHz)

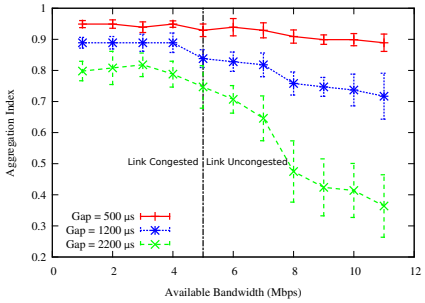

Fig. 4. AI vs. AB w.r.t. Packet Gap (802.11n,2.4GHz)

Consider the same experiment now conducted with varying levels of background traffic where the background traffic is introduced by UDP traffic from additional wireless nodes downstream from the access point. Residual capacity for each setting (1 Mb/s to 11 Mb/s) was verified to be sufficiently stable via *iperf* characterization. Figure 4 varies the available bandwidth from 1 Mb/s to 11 Mb/s and then demonstrates the resulting impact on AI as three separate spacing regimes are applied (500 microseconds, 1200 microseconds, 2200 microseconds). The packet size is varied with each of the respective spacing regimes to ensure that a constant 5 Mb/s probing rate is applied. At a spacing of 500 microseconds (0.5 ms), the aggregation index stays consistently quite high even as additional capacity opens up demonstrating limited window non-work conserving aspects of the AP. With a spacing of 1200 microseconds, the additional capacity reduces the aggregation index consistently once additional capacity is freed ($>$ 5 Mb/s, $<$ 0.9) versus the congested case ($<$ 5 Mb/s, $\geq$ 0.9). The 2200 microsecond spacing provides an exceptionally clean signal at the cost of time for the completion of the entire probing run and maximum testable rate (5 Mb/s). The key takeaway from this set of figures is that AI can be manipulated by varying rates. Hence, by introducing a variety of rates across our probing waveform or packet train, we can, therefore, manipulate the AI to determine the inflection point at which congestion occurs for helping to infer AB.

IV. FMNC Packet Train Design

The key challenge from a design standpoint is to appropriately design a timing waveform then to measure AB while capturing the AI inflection point with a relative degree of confidence and to do so with a minimal amount of time and bandwidth cost. Moreover, our packet train should be able to infer not only the AB itself but also the confidence in the final result handling cases such as upstream congestion as well as gracefully degrading in the presence of loss (see later results).

A. Available Bandwidth Train – AB

For the purposes of AB, our approach adopts a *fixed packet gap* approach rather than a fixed packet size approach as adopted by much of the prior work on AB techniques [7], [8], [15]. Thus, for the AB phase, the gap, P_G, represents the distance between the start of successive packets as sent from the FMNC server. Rate variations are achieved by varying the size of the packet subject to a maximum packet size constrained by the path MTU and allowing for sufficient inter-packet gaps to avoid the frame aggregation delay (T_{FAD}). Such constraints afford an effective range between 0 and 12 Mb/s.

Each rate is probed for $W = 20$ packets representing both the number of packets at a particular rate as well as the swap window around which TCP Sting behavior is synchronized. Each of the rate windows is oriented around a particular rate (1,3.25,5.5,7.75,10). The packet size for each rate window may then be computed by selecting a value for P_G and then computing the appropriate packet size to achieve the desired rate. An increased P_G necessitates larger packet sizes in order to accomplish the same rate within the given window as well as a longer time for completion. Given five rate windows of 20 packets within each rate window, the resulting AB test would be just over 90 KB inclusive of the Layer 3 (IPv4) and Layer 4 (TCP) headers. The packet train ($P_G \times 5 \times 20$) would then take 120,000 microseconds (120 ms) for transmission with receipt of the packets complete at the earliest at 120 ms + RTT.

B. Estimating AB

After transmission to the client, the packet train travels across the network across various bottleneck links including potentially the broadband link (link just before the AP), the WiFi link itself, as well as any other link on either the upstream or downstream link. For this section, we consider only the downstream link as a bottleneck and further revisit identification of an uplink bottleneck link in the next subsection (Section IV-C).

As noted earlier, the intuition of our approach lies in determining the threshold at which the link changes to have significant congestive effects. We identify two potential observable characteristics for determining when congestive effects have occurred:

- *High AI:* The aggregation index has risen excessively high (ex. $AI > 0.9$) likely denoting congestion on the WiFi link itself.

- *Rate mismatch:* The received rate at the client and the resulting acknowledgment rate is less or even significantly less than the probe rate, likely indicating a congested link on the path.

Based on the two criterion, given the packet sending and ACK receiving timings at the server, we are able to identify whether the sending packet(s) are congested (C) or uncongested (U). Once all the packets are tagged with either C or U, we can approximate the available bandwidth as the maximal receiving rate of the uncongested packet(s). We break down the procedure to three core steps: 1) Tagging with AI, 2) Rate Matching 3) Returning the Result.

Tagging AI: Based on our packet design, a high AI maps to a relatively certainty with regards to packet aggregation. In order to compute the AI for each received packet, we devise a metric, called *local frame aggregation*, where LAI_i is for the i-th received packet. The calculation of LAI is identical to Eq 2. The difference is that, instead of computing among all packets, LAI only computes on the subset of packets from i-th packet to $i+w$-th packet, where w is the local window size. Therefore, for an arbitrary i-th packet, $i = 0, ..., L_{AB} - w$, we can compute LAI. Then, based on the pre-defined threshold value α_{AI}, we tag all packets with $LAI_i > \alpha_{AI}$ as C. For the rest of packets, further investigation is required.

Rate matching: The sending packet rate can be easily computed with the packet size and packet gap, as observed by the server. The assumption guiding the calculation of of the receiving rate of the client at the server is that, when the uplink is not congested (ACKs are sent back from the client immediately after receiving the data packet), we can take the packet gaps of received ACKs as the equivalent of the packet gaps on the client. In the next section, we will introduce a novel technique to rule out the cases whenever an uplink bottleneck occurred.

Given the packet gap information, the challenge is to compute the receiving rate under the presence of frame aggregation. Similar to [18], we adopt the idea of a **jumbo** packet that considers the packets sent within one *MPDU* as a jumbo packet. Thus, we can compute the packet rate of the *jumbo* packet as the packet rate of each packet within this jumbo packet. If we recall the relatively stark distribution divide of packet inter-arrival times from Figure 2, a jumbo packet can be identified by packet gaps that are continuously less than T_{FAD}. Once we identify a jumbo packet, we can then compute its sending rate and receiving rate by viewing the multiple packets inside it as one large packet. Based on the receiving rate R_{rcv} and sending rate R_{snd}, we are able to tag the rest of packets with the formula[4] $R_{rcv} + \theta > R_{snd}$. If true, we tag as U, because the receiving rate is fairly high. Otherwise, we tag as C, since a decreasing receiving rate implies suppression due to congestion. At last, among all uncongested packets, we return the maximal receiving rate R_{rcv}^{max} as the input fed into the final step.

[4] $\theta = min(0.1 \times R_{snd}, 0.5)$ where R_{snd} is the sending rate. This formula helps us be fair for low rate as well as high rate.

Returning the Result: Due to our method design that focuses on specific range of rates from R_{min} to R_{max}, the final result can vary depending on the R_{rcv}^{max} that was returned from above step. We adapt the rate matching formula used above classify the final result:

- If $R_{rcv}^{max} + \theta > R_{max}$, it implies the available bandwidth is above our maximal rate (e.g., 10 Mb/s), then we classify available bandwidth as *Green*.
- Similarly, if $R_{rcv}^{max} + \theta < R_{min}$, it means the available bandwidth is less than our minimal rate (e.g., 1 Mb/s) and we classify available bandwidth as *Red*.
- Otherwise, the available bandwidth can be approximated with R_{rcv}^{max} as a explicit value from 1 to 10 Mb/s. From the perspective the classification, we call the region as *Yellow* and report an AB value. Later, we can also break down the *yellow* zone into sub-zones for judging classification accuracy.

C. Uplink Bottleneck Detection

As noted earlier, the techniques for FMNC assume that the uplink is congestion free. However, as the FMNC server only receives ACKs after a full round-trip time, congestion on the uplink can present significant ambiguity for estimation. With the completion of the AB phase and the use of TCP timestamps, we are able to successfully determine when uplink congestion has skewed the result. Although we cannot correct for uplink congestion, we can correctly identify when said congestion has occurred.

Similar to the earlier case where a local Aggregation Index AI triggered examination of the probe rate versus the ACK rate, a sufficient value of AI across a local window is then examined for uplink congestion as the root cause. Notably, TCP timestamp marking is conducted by the network stack, not the device driver. Hence, despite the fact that we do not control the actual timestamps themselves, we can use the discrepancies between stamped spacing and actual spacing to infer congestive uplink bottlenecks that resulted in frame aggregation. We use the normed difference between two adjacent packets as the metric to compute the *Pearson* correlation only when the local AI is sufficient.

With the computed correlation, where 1 implies perfect positive correlation and 0 implies no correlation, we are able to detect the cases when uplink bottleneck encountered when the correlation is well below a particular threshold PC_{Up} as the lack of correlation indicates discrepancies between the timestamp marking and the actual packet timing. For the purposes of this paper, we use $PC_{Up} = 0.55$ as derived through laboratory results.

V. Experimental Evaluation

With the various properties of FMNC now defined, we continue with controlled lab experiments to validate various aspects of the FMNC architecture. The FMNC client was written using a simple shell script (ex. *curl* requests) and was executed on a laptop running Ubuntu 14.04 with an 802.11n adapter (Ralink RT3950). The laptop, as well as the competing

background clients, were connected to a NetGear R7000 AP (802.11ac capable AP). All connections were forced to operate in 2.4 GHz to maximize congestive and interference effects though experiments were conducted in the late evening to avoid excessive non-experimental traffic. The access point was connected through a local Gigabit Ethernet switch to a computer providing NetEM-based emulation for the upstream and downstream broadband link properties. The FMNC server was run on a separate laptop using a multi-threaded *libpcap*-based C++ application and was also connected to the NetEM box via Gigabit Ethernet. Background traffic competition was provided through a mix of laptops and Raspberry Pi nodes also possessing 802.11n USB adapters (Edimax).

For the broadband link in the unconstrained case, a link capacity of 100 Mb/s was used with a 40 ms round-trip time. Background traffic through the experiment can be sourced from the server or any of the intermediate links in order to introduce congestion across the WiFi or broadband links. Background traffic was generated by the *Distributed Internet Traffic Generator* (D-ITG) [21] as we believe D-ITG allows for a finer granularity of control than *iPerf*, particularly in the lower available bandwidth ranges. Unless otherwise stated, the settings for FMNC were as follows: $T_{FAD} = 400$ microseconds,$P_G = 1200$ microseconds, 5 rate windows, 100 total packets for AB.

A. Comparing Accuracy

To start, we begin by comparing the available bandwidth accuracy computed by FMNC versus a subset of the more popular AB approaches (PathChirp [7], Spruce [8], and WBest+ [18]). We select these three approaches as PathChirp characterizes a PRM (Packet Rate Method), Spruce characterizes a PGM (Paket Gap Method), and WBest+ is aware of frame aggregation in WiFi. All three algorithms were tuned as specified in the paper (ex. Spruce was informed of the bottleneck capacity). We begin with a direct comparison by sweeping the available from 1 Mb/s to 10 Mb/s across 802.11g, 802.11n, and the broadband link where all bottlenecks are applied in the downlink case. All bottleneck values were confirmed through iperf characterization and TCP throughput testing. For the purposes of this first set of experiments, constant rate UDP packet streams were sent as cross traffic. Bottlenecks for the broadband link were applied without crossing over the WiFi link. Each method was run twenty times for each of the respective sweep values. A summary of all results can be found later in the paper in Table I.

We begin first with the comparison between the various approaches on 802.11g. Notably, 802.11g does not have frame aggregation and has only typical WiFi dynamics. As listed in Table I, the available bandwidth estimation and standard deviation of the various approaches are presented. For the results, FMNC still does an excellent job of AB prediction even without frame aggregation present demonstrating that FMNC can still use the received rate of probe packets for discerning AB. Spruce appears to fare the next best though does not fare terribly well at the lower range. However, in terms of reaction to change of available bandwidth, Spruce actually does a relatively poor job of inferring AB by continually converging to 9 Mb/s but that error is consistently better than the relatively unbounded estimations of PathChirp and WBest+. Most importantly, the coarse granularity of PathChirp and WBest+ are highly evident in the estimation errors that are consistently approaching 10Mb/s+.

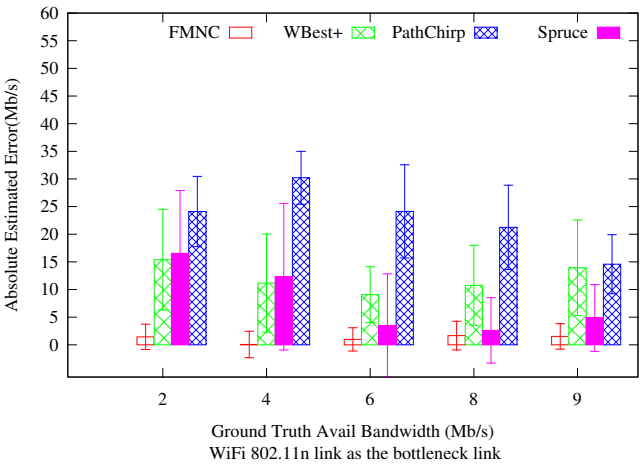

Fig. 5. Available bandwidth estimation on WiFi 802.11n link as bottleneck

Next, we continue by changing the underlying WiFi network from 802.11g to 802.11n while still keeping in the 2.4 GHz band. Figure 5 shows the estimation error for each of the four approaches. In comparison to the 802.11g case, the amount of error dramatically increases for PathChirp, Spruce, and WBest+. Out of the three comparison schemes, Spruce has the best average estimation error but that 'best' value should be tempered with the fact that the link capacity is given a priori to Spruce as required by the method. Despite knowing the link capacity, Spruce still fares poorly though in terms of the variations in its estimates. The introduction of frame aggregation with 802.11n creates no issues for FMNC which fares quite well across the entirety of the sweep.

Finally, we compare the performance of each of the three schemes when the broadband link is constrained on the downlink with a final 802.11n WiFi hop. While the error of FMNC does increase, it still fares considerably better than each of the three comparison schemes. Spruce again is the closest to FMNC but does suffer from high variations in estimation due in large part to frame aggregation issues.

One other approach to explaining the various results is to examine the various levels of aggregation present across each of the tests. Figure 6 plots the eCDF of the Aggregation Index (AI) per test across all of the tests for all of the various experiments as previously discussed in this section. Notably, 802.11g has next to zero aggregation present while the two 802.11n variations (802.11n bottleneck, 802.11g bottleneck) each have significantly higher frame aggregation present.

TABLE I
RESULTS

Bottleneck	Method	Ground Truth Bandwidth (Mbps)				
		2	4	6	8	9
802.11g	FMNC (R,G)	2.62 ± 1.78 (0.18,0.00)	4.58 ± 2.57 (0.15,0.00)	6.92 ± 2.05 (0.00,0.25)	7.60 ± 1.07 (0.00,0.35)	7.62 ± 2.14 (0.00,0.57)
	Wbest+	14.12 ± 3.04	16.73 ± 1.74	17.79 ± 1.85	19.21 ± 2.24	19.88 ± 1.70
	Spruce	8.97 ± 0.90	8.76 ± 0.80	8.83 ± 1.23	9.43 ± 0.57	10.17 ± 0.82
	PathChirp	13.96 ± 2.06	16.41 ± 1.16	17.68 ± 1.29	19.99 ± 0.92	17.27 ± 1.21
802.11n	FMNC (R,G)	3.44 ± 2.29 (0.41,0.00)	3.94 ± 2.40 (0.23,0.03)	5.01 ± 2.11 (0.03,0.03)	6.34 ± 2.60 (0.05,0.08)	8.48 ± 2.30 (0.05,0.50)
	Wbest+	17.39 ± 9.12	15.16 ± 8.88	15.07 ± 5.06	18.76 ± 7.23	23.94 ± 8.64
	Spruce	18.55 ± 11.37	16.32 ± 13.25	9.50 ± 9.33	5.39 ± 5.92	5.14 ± 6.03
	PathChirp	26.10 ± 6.35	34.21 ± 4.78	30.14 ± 8.44	29.26 ± 7.62	24.58 ± 5.34
Broadband	FMNC (R,G)	2.69 ± 1.68 (0.82,0.00)	5.12 ± 1.87 (0.41,0.05)	6.16 ± 2.72 (0.09,0.09)	6.68 ± 2.61 (0.00,0.19)	7.62 ± 1.79 (0.00,0.38)
	Wbest+	26.21 ± 4.43	28.45 ± 5.26	28.28 ± 5.75	28.28 ± 5.75	28.28 ± 5.75
	Spruce	9.02 ± 9.10	8.94 ± 8.63	8.19 ± 8.16	8.75 ± 9.51	7.05 ± 9.20
	PathChirp	21.05 ± 4.77	44.72 ± 7.46	46.80 ± 5.67	35.44 ± 7.63	36.56 ± 5.50

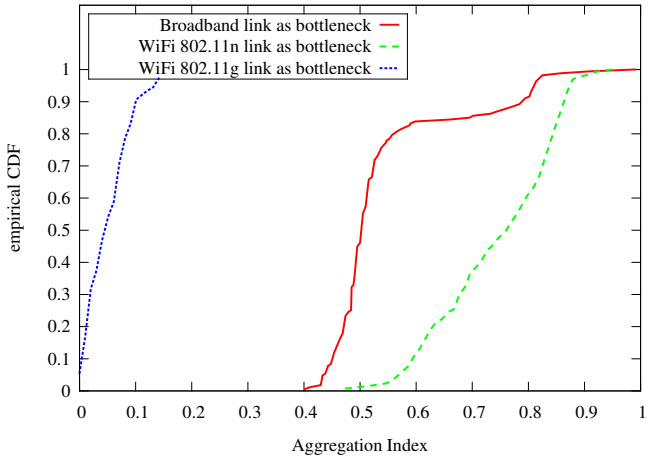

Fig. 6. The CDF of *Aggregation Index* from different bottleneck cases.

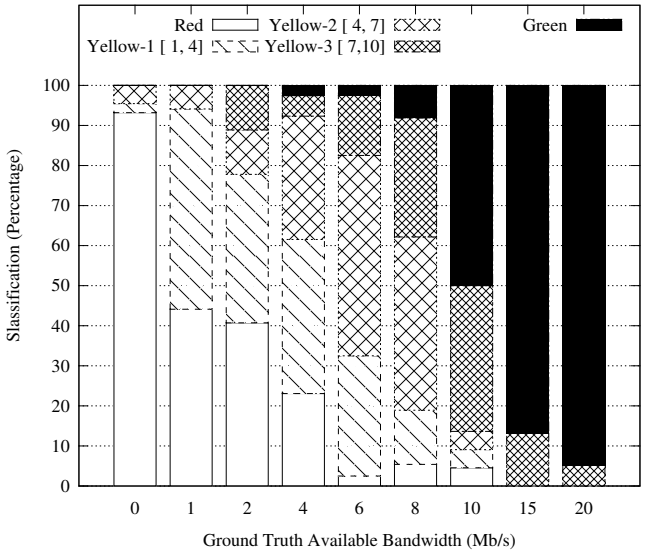

Fig. 7. FMNC AB classification v.s. Ground Truth Available Bandwidth

B. Detailed Results: AB

As noted earlier, Tables I present detailed results with respect to the performance and cost of the respective schemes. Table I breaks out the individual values for each data point from the aforementioned controlled AB tests. Each approach provides the average AB value along with the standard deviation for that particular data point. For FMNC, the result is a bit more nuanced. If FMNC detects that a result falls within the *yellow* range, an AB estimate is generated. If FMNC detects that the bandwidth is too low ($AB \max 0$), that test is classified into the *red* zone. In the table, that probability is denoted by the R value in parentheses under the *yellow* zone estimation. Similarly, for cases where FMNC detects that the likely bandwidth is greater than 10 Mb/s, that test is marked as *green*. That probability is reported in the G slot in parentheses.

C. Additional Evaluations

Tri-color classification: To expand upon the tri-color classification of FMNC, Figure 7 presents the classification result as the bandwidth is varied from $AB = 0$ to $AB = 20$ Mb/s. For the purposes of improved discernment, the yellow region is broken down into three categories, Yellow 1 through Yellow 3 with reach representing a particular range of bandwidth. For the user perspective and the server perspective, the goal is either to outright classify the bandwidth correctly or to only slightly miss in terms of the bandwidth classification knowing that short-term wireless dynamics may create temporary uncertainty. Notably, FMNC fares quite well in terms of classification accuracy which for all practical purposes is the most important takeaway for the user.

Uplink congestion: To test the uplink congestion, we congest the more difficult (for FMNC) broadband link using UDP traffic. In Figure 8, the uplink utilization is varied from 0 to 100% with both the Aggregation Index (for the entire test) and the Pearson correlation (entire test) also plotted. Notably, the Pearson correlation hovers between 0.5 to 0.7 before drastically dropping to nearly 0.2 when the uplink is

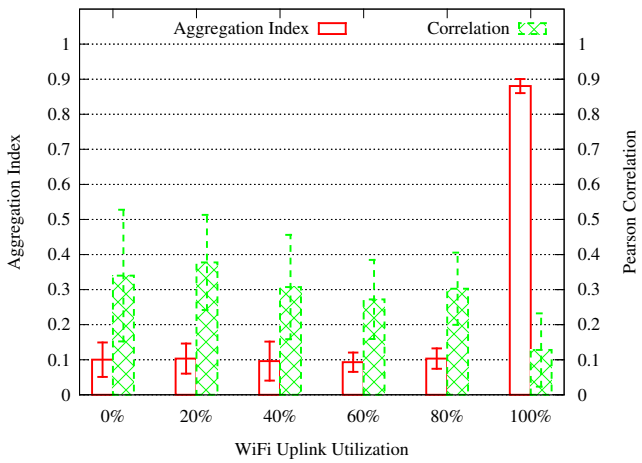

Fig. 8. Uplink congestion detection

congested as predicted by FMNC.

VI. Summary

In conclusion, we presented Fast Mobile Network Characterization in this paper, a technique capable of rapid, lightweight, and accurate WiFi path characterization. We showed in our paper how we can leverage the detection of aggregation in the reaction of probe packets in order to rapidly characterize the available bandwidth. Through small, sliced, and reordered packets, we demonstrated how intelligent packet train construction can accurately characterize from 0 to 10 Mb/s, the area of most relevance for user QoE. We showed through both lab experiments as well as larger real-world studies the extent by which FMNC can yield accurate and intriguing insight. Our future efforts include extending FMNC to cellular networks and exploring the scaling properties of FMNC.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. CNS-1718405 and Grant No. CNS-1439682.

References

[1] iperf3. software.es.net/iperf. [Online]. Available: software.es.net/iperf
[2] Speed test. www.speedtest.net. [Online]. Available: www.speedtest.net
[3] A. Nikravesh, H. Yao, S. Xu, D. Choffnes, and Z. M. Mao, "Mobilyzer: An open platform for controllable mobile network measurements," ser. MobiSys '15. New York, NY, USA: ACM, 2015, pp. 389–404. [Online]. Available: http://doi.acm.org/10.1145/2742647.2742670
[4] S. Rosen, S. j. Lee, J. Lee, P. Congdon, Z. M. Mao, and K. Burden, "Mcnet: Crowdsourcing wireless performance measurements through the eyes of mobile devices," *IEEE Communications Magazine*, vol. 52, no. 10, pp. 86–91, October 2014.
[5] W. Wei, S. Jaiswal, J. Kurose, D. Towsley, K. Suh, and B. Wang, "Identifying 802.11 Traffic From Passive Measurements Using Iterative Bayesian Inference," *IEEE/ACM Transactions on Networking*, vol. 20, no. 2, pp. 325–338, Apr. 2012. [Online]. Available: http://dl.acm.org/citation.cfm?id=2317330.2317331
[6] J. Sommers and P. Barford, "Cell vs. WiFi," in *IMC '12*. New York, New York, USA: ACM Press, nov 2012, p. 301. [Online]. Available: http://dl.acm.org/citation.cfm?id=2398776.2398808
[7] V. J. Ribeiro, J. Navratil, R. H. Riedi, R. G. Baraniuk, and L. Cottrell, "pathchirp: Efficient available bandwidth estimation for network paths," in *Presented at*, no. SLAC-PUB-9732, 2003.
[8] J. Strauss, D. Katabi, and F. Kaashoek, "A measurement study of available bandwidth estimation tools," in *IMC '03*. New York, New York, USA: ACM Press, Oct. 2003, p. 39. [Online]. Available: http://dl.acm.org/citation.cfm?id=948205.948211
[9] M. Li, M. Claypool, and R. Kinicki, "Wbest: A bandwidth estimation tool for ieee 802.11 wireless networks," in *Local Computer Networks, 2008. LCN 2008. 33rd IEEE Conference on*. IEEE, 2008, pp. 374–381.
[10] C. D. Mano, A. Blaich, Q. Liao, Y. Jiang, D. A. Cieslak, D. C. Salyers, and A. Striegel, "RIPPS," *ACM Transactions on Information and System Security*, vol. 11, no. 2, pp. 1–23, May 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1330332.1330334
[11] S. Savage, "Sting: A tcp-based network measurement tool." in *USENIX Symposium on Internet Technologies and Systems*, vol. 2, 1999, pp. 7–7.
[12] R. Kapoor, L.-J. Chen, L. Lao, M. Gerla, and M. Y. Sanadidi, "CapProbe," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, p. 67, oct 2004. [Online]. Available: http://dl.acm.org/citation.cfm?id=1030194.1015476
[13] R. L. Carter and M. E. Crovella, "Measuring bottleneck link speed in packet-switched networks," *Perform. Eval.*, vol. 27-28, pp. 297–318, Oct. 1996. [Online]. Available: http://dx.doi.org/10.1016/0166-5316(96)00036-3
[14] M. Jain and C. Dovrolis, "Pathload: A measurement tool for end-to-end available bandwidth," in *In Proceedings of Passive and Active Measurements (PAM) Workshop*, 2002, pp. 14–25.
[15] P. Steenkiste, "Evaluation and characterization of available bandwidth probing techniques," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 6, pp. 879–894, Aug. 2003. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1217275
[16] B. Melander, M. Bjorkman, and P. Gunningberg, "A new end-to-end probing and analysis method for estimating bandwidth bottlenecks," in *Globecom '00*, vol. 1. IEEE, 2000, pp. 415–420. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=892039
[17] J. Navratil and R. L. Cottrell, "Abwe: A practical approach to available bandwidth estimation," in *in Passive and Active Measurement (PAM) Workshop 2003 Proceedings, La Jolla*, 2003.
[18] A. Farshad, M. Lee, M. K. Marina, and F. Garcia, "On the impact of 802.11n frame aggregation on end-to-end available bandwidth estimation," in *SECON*. IEEE, Jun. 2014, pp. 108–116. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6990333
[19] A. Farshad, M. K. Marina, and F. Garcia, "Urban wifi characterization via mobile crowdsensing," in *Network Operations and Management Symposium (NOMS), 2014 IEEE*, May 2014, pp. 1–9.
[20] S. Deng, R. Netravali, A. Sivaraman, and H. Balakrishnan, "WiFi, LTE, or Both?" in *IMC '14*. New York, New York, USA: ACM Press, nov 2014, pp. 181–194. [Online]. Available: http://dl.acm.org/citation.cfm?id=2663716.2663727
[21] A. Botta, A. Dainotti, and A. Pescapè, "A tool for the generation of realistic network workload for emerging networking scenarios," *Computer Networks*, vol. 56, no. 15, pp. 3531–3547, 2012.