FAM: A frame aggregation based method to infer the load level in IEEE 802.11 networks

Nour El Houda Bouzouita, Anthony Busson, Hervé Rivano

FAM: A Frame Aggregation based Method to infer the load level in IEEE 802.11 networks*

Nour El Houda Bouzouita†, Anthony Busson‡and Hervé Rivano§

28 April 2022

Abstract

In many environments, connected devices are exposed to and must choose between multiple Wi-Fi networks. However, the procedure for selecting an access point is still based on simple criteria that consider the device to be unique in the network. In particular, the network load is not taken into account even though it is a key parameter for the quality of service and experience.

In this paper, we investigate how an unmodified vanilla device could estimate the load of a network in the user space with no interventions from the access points. In this regard, we propose a novel and practical method, FAM (**F**rame **A**ggregation based **M**ethod). It leverages the frame aggregation mechanism introduced in recent IEEE 802.11 amendments to estimate the network load through its channel busy time fraction. FAM combines an active probing technique to measure the actual packet aggregation and Markovian models that provide the expected rate as a function of the volume and nature of the traffic on the network. We validate the effectiveness of FAM against both ns-3 simulations and test-bed experiments under several scenarios. Results show that our method FAM is able to infer the network load with a granularity based on six different levels of network loads for the considered scenarios.

1 Introduction

Wireless Local Area Networks (WLANs), especially IEEE 802.11, are widely deployed in a variety of situations: homes, corporate or campus networks, public areas, etc. [1]. To join a network, a user terminal has to associate with an

*Article published in Elsevier Computer Communication https://doi.org/10.1016/j.comcom.2022.04.021

†Corresponding author, nour-el-houda.bouzouita@ens-lyon.fr, Univ Lyon, UCBL, EnsL, CNRS, Inria, LIP, F-69342, LYON Cedex 07, France.

‡anthony.busson@ens-lyon.fr, Univ Lyon, UCBL, EnsL, CNRS, Inria, LIP, F-69342, LYON Cedex 07, France.

§Herve.Rivano@inria.fr, Univ Lyon, INSA Lyon, Inria, CITI, EA3720, 69621 Villeurbanne, France

access point (AP). In many cases, the choice of the access point to associate with is trivial, there is only one, or controlled: the objective is to join a specific network that is composed of several access points belonging to the same Extended Service Set and coordinated by a centralized WLAN controller. This controller can implement performance optimization algorithms such as channel allocation, load balancing, etc [2, 3] and guide the device to a chosen AP.

In other cases, especially in public areas, there may be the choice between several access points, potentially belonging to different networks, without any coordination between them. If we take the example of a train station, a Wi-Fi network with a couple of APs may have been deployed by the train's company to offer free Wi-Fi service. Shops, bars, and restaurants may have deployed their own Wi-Fi networks, each composed of one or several independent APs. These public networks are managed by different entities. There is no common policy or rational management to help the devices selecting the Wi-Fi network or AP that offers the best performance.

It is then up to the terminal to choose the access point it will associate with.

Despite its importance, the choice made by the operating system is often based on simple criteria considering that the device is alone in its vicinity. Usual metrics focus on the received signal strength indicator (RSSI), if not only on a static list established from the user's connection history. In particular, these metrics do not relate to the quality of service and experience that the device will experience.

The question of interest that we pose in this paper is: how could a device choose an access point to associate with based on the expected network performances? Unfortunately, classical performance metrics such as the average throughput cannot be easily forecast by a device. Indeed, the available bandwidth as measured by a given station depends on many factors, including the antenna and channel gains that are approximately captured by the RSSI, but also many other hardware or software parameters that condition the modulation and coding schemes it can use. The network load and the characteristics of the competing traffic are also crucial.

The network load can be expressed in many ways. In this work, we consider the Busy Time Fraction (BTF), defined as the fraction of time the wireless medium is sensed busy due to successful or unsuccessful transmissions. It captures concurrent transmissions, constituting the AP load, as well as inter-network interference. Some specific hardware can provide this quantity. In particular, it may be included in IEEE 802.11k measurement reports, but this is only available to the APs implementing this amendment and not provided to the end user device. It is also possible, under some hardware conditions, for a computer to switch its Wi-Fi card in monitor mode and obtain the BTF. Unfortunately, it requires privileged access and not all drivers are offering this functionality. In particular, this is not applicable to non-rooted handheld devices with traditional operating systems.

The contribution of this paper is to evaluate the possibility/capacity for a

vanilla Wi-Fi client, typically a smartphone, to infer the Wi-Fi network load from local measurements in the user space. To circumvent the aforesaid limitations, we consider active probing of the network, which has been widely used in the literature to estimate network loads and which is considered to be an effective method for providing valuable insights into network status [4, 5, 6].

We model, simulate, and experiment on a test-bed, a method that allows a client associated with a Wi-Fi network to estimate the BTF of the network. It uses a server that can be another Wi-Fi client associated with the same AP (e.g., another smartphone) or a server on the wired network. For the latter, the method assumes that the bottleneck is the Wi-Fi network. The method constitutes a proof of concept that shows that the load is inferable locally when the server is close to the client, without any control on the AP. The method is not yet applicable to the case where the server is arbitrarily located on the Internet. In order to achieve this definitive application, other scientific challenges need to be tackled, such as the modeling of the impact of Internet routing on traffic. This is, however, out of the scope of this paper.

The proposed method is named FAM (**F**rame **A**ggregation based **M**ethod). It leverages the frame aggregation mechanism, introduced since the IEEE 802.11n amendment, to estimate the channel busy time fraction, thus the current load. Moreover, our test-bed experiences and simulations have shown that the throughput that a joining device could get depends on whether the competing traffic uses frame aggregation or not. Therefore, FAM estimates not only the BTF but also the nature of the traffic in the following way. Thus, making it possible to evaluate the available bandwidth that another device would have if it would join the network.

- FAM launches an active probing of the network and measures the aggregation level of its transmissions. The number of probes is a trade-off between cost and accuracy computed using the central limit theorem.
- We propose several Markov models providing the expected aggregation levels of the device given the characteristics of the competing traffic, its intensity, and its use of frame aggregation.
- FAM compares the behavior of the actual aggregation levels to the expected ones and identifies the BTF and the nature of the traffic on the network.

FAM accuracy is evaluated through extensive discrete-event simulations and test-bed experiments. From the results, we can get the following engineering insights on the feasibility of a crowd-sourcing platform for network load estimation.

- The accuracy of FAM is enough for the classification of the load in few levels.
- When the majority of the competing traffic does not use frame aggregation, it is only possible to identify two classes.

- The number of devices composing the DownLink competing traffic which is coming from the same AP does not influence the prediction.

The remainder of this paper is organized as follows. Related works are discussed in Section 2. Section 3 describes, in detail, the frame aggregation and block acknowledgment schemes. In Section 4, we describe the considered system. Section 5 presents our proposed analytical models to estimate the frame aggregation level. Our estimation method, FAM, is presented in Section 6. Section 7 is dedicated to the numerical validation of our proposed models and method FAM. Finally, Section 9 concludes this paper.

2 State of the Art

As a crucial metric for capturing the network state, available bandwidth has become a key parameter for measuring the overall network performance. Available bandwidth estimation tools can be broadly categorized as active and passive measurements. Passive approaches [7, 8] evaluate the available bandwidth by monitoring the network traffic without emitting any traffic. They often need specific hardware, e.g., chipsets that can switch to monitor mode and dedicated capture software. Some techniques even need to be run on routers. On the contrary, active methods often need lighter set-up and are easier to deploy and run on regular devices. The counterpart is that they inject probe traffic in the network, thus perturbing the measure itself.

Multiple active measurement techniques have been developed, in the field of the estimation of available bandwidth and busy time, in wired and wireless networks. Generally, most of these works can be grouped into two classes.

- **The Packet Rate Model (PRM):** This model is based on the concept of self-induced congestion. It calculates the end-to-end available bandwidth by monitoring the probe and received packet rates and detecting the queuing delays. For doing so, a sequence of small probe packets are sent at different rates from a source to a sink node. If the probe packet rate exceeds the actual available bandwidth, then probe packets will be queued up at a router, and the packet rate at reception will be therefore lower than at emission. Thus, the available bandwidth can be discerned by detecting the turning rate at which queuing delays start to occur.

  A significant number of PRM techniques have been developed, such as TOPP [9], pathload [10], pathChirp [11] and DietTOPP [12]. These tools differ according to the probing rate adjustment and in their receiver-side analysis approaches. Pathload uses Constant Bit Rate (CBR) streams and adjusts the probe rate during each round based on a binary search method. TOPP relies on a linearly growing rate. As a typical PRM technique, DietTOPP deploys the TOPP algorithm with a simplified search method. Compared to Pathload, PathChirp reduces probe traffic overhead by using a sequence of exponentially spaced probe packets of the

same size, denoted chirps. Within the same chirp, several rates can be probed, thereby improving accuracy.

- **The Probe Gap Model (PGM):** This model measures the available bandwidth by inferring the intensity of the cross traffic at the bottleneck. PGM techniques typically send a batch of back-to-back probes into the network path at a single rate and rely on the dispersion in time between two successive probes at the receiver side to estimate the cross traffic. The PGM approaches, such as Spruce [13] and Initial Gap Increase/Packet Transmission Rate (IGI/PTR) [14] send batches of probe packets and measure the inter-arrival time of the consecutive packets. Since the contention with the cross traffic induces queuing delays, the packets will be dispersed in time. This dispersion increases with the load, thus making it possible to infer the available bandwidth. Delphi [15] probes the path with a series of chirp trains. Based on the transmitted and the received inter-packet delays relationship, it can estimates the load induced by the cross traffic.

Most of the aforesaid techniques have been designed for classic wired networks. The rise of wireless networks and complex network infrastructures have fostered the development of available bandwidth estimation approaches, e.g., for 3G/4G networks [16], cloud networks [17], ad-hoc networks [18, 19], or Software Defined Networking (SDN) networks [20]. The advent of these new tools has motivated some studies, such as [21] to propose an updated summary of the metrics, characteristics, and techniques related to the measurement of the available bandwidth.

The methods developed for wired networks fail when applied to Wi-Fi, in particular because of the CSMA/CA MAC method that implies complex waiting times, e.g., random backoff, Distributed Inter-Frame Space period (DIFS), etc. The mechanisms used to cope with the fluctuating wireless channel conditions, e.g., interference, fading, and bit error rates, need also a specific attention: dynamic rate adaptation, Automatic Repeat Request (ARQ).

Available bandwidth estimation in IEEE 802.11 networks In [22], the authors proposed IdleGap, which requires to run on a real-time system. It infers the idle time fraction, defined as the fraction of time in which the channel is idle, based on low layer information, such as the network allocation vector (NAV). Such information is generally not available at the upper layer making this technique difficult to implement in practice. WBest [4] is a PGM tool consisting of two steps. In the first step, it sends pairs of probe packets and infers the effective capacity of the network, defined as the maximum capability of the wireless network to deliver network layer traffic [23]. It then transmits a probe packet train at the estimated rate and infers the available bandwidth based on the dispersion rate.

All cited approaches were proposed before the implementation of the frame aggregation scheme (detailed in the following section) by the recent IEEE 802.11

standards, making them ill-suited for deployment. Indeed at the application level, aggregated frames are received simultaneously. PGM techniques cannot process the negligible gap between packets that were aggregated in the same frame, and PRM schemes may distort and overestimate or underestimate the received rate.

Aggregation aware available bandwidth estimation Recent works have adapted PGM and PRM techniques to take into account frame aggregation. In [6], the authors proposed WBest+ that deploys the WBest algorithm with a modified packet rate calculation method by considering aggregated frames as a unique jumbo frame. The available bandwidth estimation is performed based on the time between aggregated frames instead of the time between probe packets. Another solution is proposed by L. Song and A. Striegel [5], who have proposed Aggregation Intensity based WiFi Characterization (AIWC), aiming at estimating the frame aggregation level at the reception side to capture link congestion and deduce the available bandwidth. To capture the aggregation, AIWC and WBest+ rely on a threshold-based method: if the gap time between two consecutive received packets is less than a given threshold, the corresponding packets are deemed to be aggregated. The network scenario used in these papers consists of a server with a wired Internet connection that sends the probing traffic along the network path from the server to the client with a last-hop wireless connection that is assumed to be the bottleneck. The scenario considered in this paper is different as the probe traffic is sent from the client to the server.

Analytical models At the other end of the literature, there are the modeling approaches that analyze the performance of the MAC Frame aggregation techniques based on analytical models. Hajlaoui et al. proposed a Discrete-Time Markov Chain (DTMC) to model the functioning of the aggregation mechanism and block acknowledgment under the assumption of a binary symmetric channel [24]. Then, they presented an analytical model to evaluate IEEE 802.11n saturation throughput based on the proposed DTMC. A property of this model is that it considers saturated networks. Because the saturation assumption can be deemed too restrictive in some cases, in [25], B. S. Kim et al. proposed an enhanced DTMC to study the frame aggregation post-backoff and evaluate the throughput performance under unsaturated conditions. However, the objectives of these works differ from ours and the scenarios considered in these papers are not suitable for available bandwidth estimation as they require different traffic patterns.

In this paper, we propose a relatively simple and versatile analytical model specific to the application of BTF estimation in presence of the IEEE 802.11 frame aggregation scheme. We model and simulate scenarios in which a device induces the mean aggregation level of an aggregated deterministic probe traffic competing with a cross traffic that can aggregate or not its frames. The analysis results are then delivered to our scheme FAM to characterize the channel load and the cross traffic type.

3 New IEEE 802.11 features overview

The recently rewritten IEEE 802.11 standard [26] and the most recent amendments, IEEE 802.11n, IEEE 802.11ac, and IEEE 802.11ax, brought in significant network efficiency and channel utilization gains by introducing new PHY and MAC layer features. The improvement of the MAC layer, since 802.11n, relies mainly on two new mechanisms, Frame Aggregation and Block Acknowledgment (Block ACK). In the following, we briefly describe these two enhancements.

3.1 Frame aggregation

When frame aggregation is enabled, several sub-frames are concatenated into a single large frame before transmission, thereby reducing MAC layer overhead. There are two types of frame aggregation: Aggregate MAC Protocol Data Unit (A-MPDU) and Aggregate MAC Service Data Unit (A-MSDU). These two schemes differ by wherein the network stack they apply the frame aggregation. A-MSDU is performed before the MAC header encapsulation process. It aggregates multiple Service Data Units (SDUs) with one common MAC header and sends them with a common PHY header to the same receiver. As a consequence, the corruption of any sub-frame during transmission causes the corruption of the whole aggregated frame. To our knowledge, A-MSDU is rarely implemented in practice. In this paper, we focus on A-MPDU because it is mandatory in the standard and implemented by default in recent Wi-Fi cards.

The A-MPDU frame aggregation is performed after the MAC header encapsulation process, where several MAC protocol data units (MPDUs) are aggregated together in a single PHY protocol data unit (PPDU) frame with a common PHY header and send it to the same receiver. Each A-MPDU, as illustrated in Figure 1, starts with an MPDU delimiter followed by the MPDU, consisting of its own MAC header, MAC payload, and FCS, and ends with padding bytes. Consequently, the corruption of any A-MPDU sub-frame does not cause the corruption of the whole A-MPDU, and only the corrupted MPDUs must be transmitted again.

Note that the maximum number of frames assembled within the same A-MPDU depends on three factors:

- The block ACK frame can acknowledge a maximum of 64 frames (the 802.11ax amendment increased it to 256).
- The size of an A-MPDU is limited and depends on the Wi-Fi card vendor's implementation.
- The A-MPDU has a maximum transmission duration, which depends on the data transmission rate.

The frame aggregation level is therefore dependent on the sender buffer state at the moment it accesses the medium.

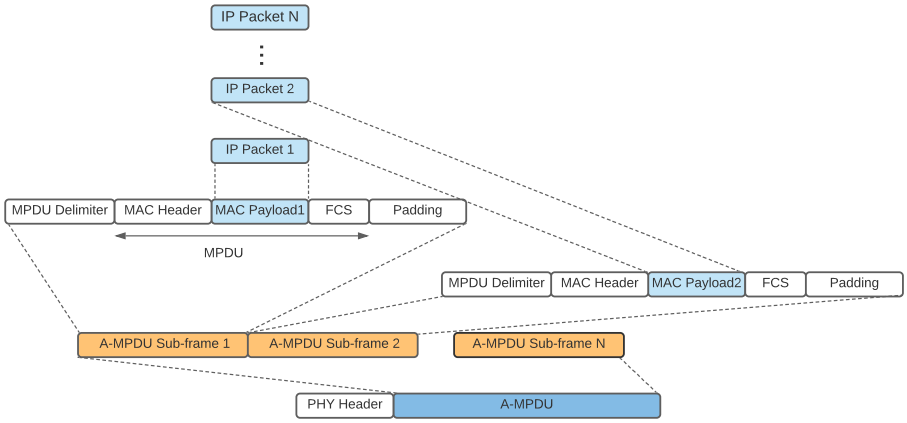


Figure 1: A-MPDU frame aggregation

3.2 Block acknowledgement

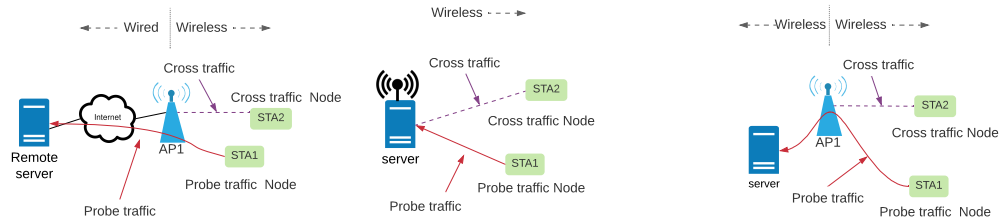
IEEE 802.11 frames have to be acknowledged. When multiple data frames are assembled into one A-MPDU, sending a single positive acknowledgment for each A-MPDU sub-frame would not consolidate the frame aggregation efficiency gains further. Therefore, when the receiver receives an A-MPDU, it sends back a unique Block ACK that contains a bitmap filed indicating explicitly which sub-frames have been received properly and which sub-frames have been corrupted. The Block ACK acknowledges the correctly received frames and requests to retransmit the corrupted frames. There are two Block ACK schemes: immediate and delayed. With the immediate Block ACK, the receiver sends the acknowledgment right after the reception of the transmitted frames, while with the delayed Block ACK, the receiver can wait before acknowledging. Delayed Block ACK is appropriate for applications that support moderate latency traffic, whereas immediate Block ACK is suitable for applications that require low latency and high bandwidth.

4 System description

The system we consider is a general WLAN that uses the IEEE 802.11 Distributed Coordination Function (DCF). Figure 2 depicts the overall architecture.

The user device is denoted the probe node. It associates to $AP1$ and sends a probe traffic to the server. We present three hypothesis on the server's position inside the network's topology.

Figure 2a presents a scenario similar to classical "Speed Test" applications where the server is located outside the local networks of the Wi-Fi APs. How-



(a) scenario 1

(b) Scenario 2

(c) Scenario 3

Figure 2: BTF measurement architecture

ever, the characteristics of the connection within the Internet between the AP and the server are too complex to capture and model. We therefore introduce the scenario in Figure 2b in which we assume that the connection between the AP and the server is ideal. We model it as if the server were implemented on the AP. The scenario of Figure 2c is a more realistic and practical one. We consider that the server is embedded on a second wireless device, owned by the user and connected to the same AP through the same Wi-Fi network. In the following, we model, simulate and implement the second and third scenarios.

The probe node and the server run an application that aims to infer the two following informations.

- The load of the channel. This load can be generated by the traffic from $AP1$ or other APs/stations (STAs) using the same channel. The load is defined here as the BTF.
- The type of traffic: do other nodes aggregate their frames or not?

The traffic carried by the network is denoted the *cross traffic*. We consider that the downlink (DL) cross traffic is the predominant compared to the uplink (UL) traffic. When the probing application runs, it generates a supplementary traffic denoted the *probe traffic*. It's an uplink traffic from the probe node to the AP. In the wireless server scenario, it is also downlink.

To perform the BTF estimation and the type of concurrent traffic, the probe node sends a sequence of small probe packets to the server with an increasing inter-packet arrival using the User Datagram Protocol (UDP). The probing node is assumed to aggregate frames and thus uses a recent IEEE 802.11 amendment (since IEEE 802.11n). Upon receiving the probe packets, the server analyzes them with the algorithm given in Section 6 and deduces the BTF and the presence of aggregation. This estimation relies on an analytical modeling approach described in Section 5. Based on the scenario used, we propose two models. The first model, called "Ideal server", simulates the simplified case where the server is implemented on the AP. The second model is called "Wireless server" and captures the case where the server is embedded on a device associated with

the AP.The frame aggregation level is computed using the corresponding model in each case.

5 Analytical models and their solutions

In this section, the description of the two modeling approaches is introduced. We start by describing the common assumptions used in the two models and the notations used for their mathematical formulations. The system we consider is a general IEEE 802.11 infrastructure-based WLAN configured to use the DCF access method. We distinguish two different entities: an AP and stations that can be either user stations or servers.

The two proposed analytical models are Discrete-Time Markov Chains (DTMC). They evaluate the aggregation levels of the probe traffic for a given cross traffic load. While the first model estimates the probe aggregation level of the UL traffic sent by the probe node, the second model appraises the aggregation level of the DL traffic forwarded by the AP to the server.

As the probe frame aggregation level depends on the nature of cross traffic, each model relies on two Markov chains. The first chain considers that the cross traffic uses frame aggregation, whereas the second is based on non-aggregated cross traffic. Table 1 summarizes the principal notation used in the paper.

Our models rely on the following assumptions:

- **Probe traffic**: It is a Constant Bit Rate (CBR) traffic, sent at regular interval d_p. Frame aggregation scheme is always enabled for this flow.
- **Cross traffic**: The cross traffic is modeled by a constant bit rate source sending packets at regular interval d_c and managed by a unique queue. The cross traffic can be either aggregated or not. This flow is coming from the distribution system (a wired network connected to the AP, which is not represented in Figures 2b and 2c) and sent to the cross traffic node.
- **Buffers**: The probe traffic node, the cross traffic node, and the AP buffers are assumed to have a finite size. More precisely, we assume that when an aggregated frame is sent for a given destination, the corresponding buffer becomes empty for this destination. Buffer thereby becomes empty after each transmission.

The impact of these assumptions is evaluated through simulations and test-bed experiments and discussed in Section 7.

5.1 Ideal server model

Let us first consider the "ideal server" model that captures the scenario of Figure 2b. There 3 nodes: an AP, and two user stations. The first sends the probe packets to the server located on the AP. The second receives the cross traffic forwarded by the AP. As mentioned earlier, we propose two Markov chains for each model. For the sake of readability, we describe only the second chain

Table 1: Principal notation

Parameter (unit)	Definition
T_{DIFS} (μs)	Distributed Inter Frame Space
T_{SIFS} (μs)	Short Inter Frame Space
T_{PHY} (μs)	The preamble and physical header duration
FCS ($Bytes$)	Frame Check Sequence
$T_{BlockACK}$ (μs)	Required time to send the block acknowledgment
$T_{backoff}$ (μs)	Average backoff time
T_{slot} (μs)	Slot time
CW_{min}	Minimum size of the contention window
d_c (μs)	Inter-arrival time of cross traffic packets
d_p (μs)	Inter-arrival time of probe traffic packets
$AMPDU_{AP}$	Maximum A-MPDU size for the AP
$AMPDU_P$	Maximum A-MPDU size for the probe traffic node
Ideal server model	
$P_{(i,j)(l,m)}$	Transition probability from state (l,m) to state (i,j)
$f(k)$ (μs)	Required time to send k probe aggregated sub-frames
g (μs)	Required time to send a single cross traffic frame
Wireless server model	
$P_{(i,j,k,u)(l,m,q,v)}$	Transition probability from state (i,j,k,u) to state (l,m,q,v)
$T_{AP}(k)$ (μs)	Required time to send k DL probe aggregated sub-frames
$T_{SP}(k)$ (μs)	Required time to send k UL probe aggregated sub-frames
$T_{AC}(k)$ (μs)	Required time to send k DL cross traffic aggregated sub-frames

of this model where the frame aggregation mechanism is disabled for the cross traffic. The aggregated version is presented and analyzed in [27]. This scenario permits us to evaluate the aggregation level of the UL probe traffic sent by the probe traffic node.

We consider the Markov chain defined as the couple $(X_n, Y_n)_{n\geq 0}$. X_n describes the number of aggregated sub-frames contained in the n^{th} transmitted probe frame, while Y_n represents the number of packets at the cross traffic buffer at the moment of the n^{th} probe frame transmission. The set of all possible states is $\{0, ..., AMPDU_P\}$ for the X_n process and $\{0, ..., AMPDU_{AP}\}$ for the Y_n process.

Transition probabilities The transition probabilities $P_{(i,j)(l,m)}$ are determined by the time between two consecutive probe traffic transmissions. As both probe and cross traffics are deterministic, this time sets the number of packets that arrived in the two buffers between two transmissions and thus the number of frames that will be sent in the aggregated frame. Consequently, we analyze the events that may occur between two probe traffic transmissions.

For the sake of clarity, we recourse to Figure 3 to describe the execution steps of the proposed model. It shows an example of possible events between

two successive probe transmissions when the aggregated probe traffic and the non-aggregated cross traffic are competing for the channel resource.

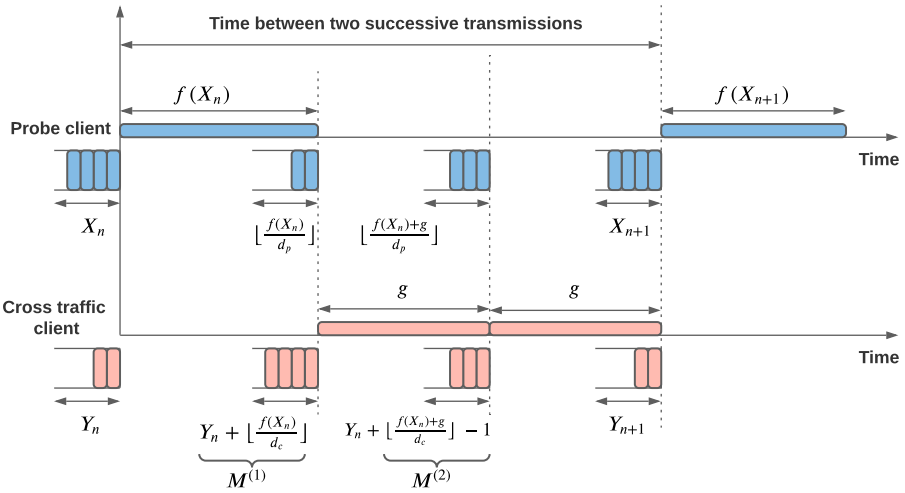


Figure 3: Example timeline of possible events between two successive probe transmissions when the cross traffic does not aggregate its frames.

In this example, at time 0, we start from a state (l, m), i.e., $(X_n = l, Y_n = m)$ $((4, 2)$ in Figure 3). By construction, a transition begins by a probe transmission. It sends the l frames currently in its buffer, aggregating them in a unique A-MPDU (4 in our example) using a specific data rate. The transmission duration is denoted $f(l)$ which is given in A. During this transmission, the probe buffer receives $\lfloor \frac{f(l)}{d_p} \rfloor$ packets and the cross traffic buffer receives $\lfloor \frac{f(l)}{d_c} \rfloor$. Consequently, at the end of this transmission, the probe and cross buffers will contain respectively, $\lfloor \frac{f(l)}{d_p} \rfloor$ and $m + \lfloor \frac{f(l)}{d_c} \rfloor$ packets (2 and 4 in our example).

Before the next probe frame transmission, an arbitrary number k of successive transmissions of cross traffic can occur. In our example, the cross traffic gains two successive medium accesses. We denote $M^{(k)}$ the number of frames in the cross traffic buffer before its k^{th} transmission. $M^{(1)}$ is equal to $m + \lfloor \frac{f(l)}{d_c} \rfloor$. As cross traffic is not aggregated, a single frame among the $M^{(1)}$ is sent. We denote this frame duration by g. It differs from $f(.)$ as the local properties (transmission rates, packet sizes) and the protocol are different (IEEE 802.11g, for instance) for the cross traffic. At the end of the first cross traffic transmission, the cross buffer contains $M^{(2)}$ frames which can be calculated by $M^{(1)} - 1 + \lfloor \frac{g}{d_c} \rfloor$. Generalizing for $k > 1$, we get:

$$M^{(k)} = M^{(k-1)} - 1 + \left\lfloor \frac{g}{d_c} \right\rfloor \tag{1}$$

We define $Q(l,m)$ the random variable that describes the number of consecutive times that the cross traffic gains access to the medium.

Besides, we denote by $p(k)$ the probability for the cross traffic to access the medium at least k successive times given that probe and cross traffic have non-empty buffers. This probability depends on the IEEE 802.11 amendment used by the cross and probe traffic (size of the contention window, etc.).

We distinguish the case where $Q(m,l) = 0$:

$$\mathbb{P}\Big(Q(l,m) = 0\Big) = \mathbb{1}_{m+\frac{f(l)}{dc}<1} + (1 - p(1)) \cdot \mathbb{1}_{m+\frac{f(l)}{dc}\geq 1} \tag{2}$$

The first term corresponds to the case where the cross traffic buffer is empty, and the second term where the cross traffic node loses access to the medium at its first attempt.

$$\mathbb{P}\Big(Q(l,m) = k\Big) = \prod_{q=1}^{k} \mathbb{1}_{M^{(q)}\geq 1} \cdot \Big(p(k)\mathbb{1}_{M^{(k+1)}=0} + (p(k) - p(k+1))\mathbb{1}_{M^{(k+1)}>0}\Big) \tag{3}$$

In this equation, the product corresponds to the probability that the cross traffic has a non-empty buffer during each of the k successive transmissions. The term in brackets describes the probability that the cross traffic does not access the medium after its k^{th} transmission, either because it loses access to the medium at the $k+1$ accesses when competing with the probe traffic (where the term $(p(k) - p(k+1))$ is a factor) or because of an empty buffer. Since two types of traffics, the probe and the cross traffic, are competing with a $\frac{1}{2}$ chance of accessing the medium each, setting $p(k) = \frac{1}{2^k}$ is a relevant choice for the numerical application.

We are now able to calculate the transition probabilities based on the number of cross traffic accesses and their transmission times. As the probe traffic is deterministic, the number of frames at the probe traffic buffer is thereby fully determined from this time.

For $i \geq 2$, we get:

$$P_{(i,j)(l,m)} = \sum_{k=0}^{\infty} \mathbb{P}\Big(Q(l,m) = k\Big) \cdot \mathbb{1}_{d_p \cdot i \leq f(l) + k \cdot g < d_p \cdot (i+1)} \cdot \mathbb{1}_{M^{(k+1)}=j} \tag{4}$$

For $i = 1$, we get:

$$P_{(i,j)(l,m)} = \sum_{k=0}^{\infty} \mathbb{P}\Big(Q(l,m) = k\Big) \cdot \mathbb{1}_{0 \leq f(l) + k \cdot g < 2d_p} \cdot \mathbb{1}_{M^{(k+1)}=j} \tag{5}$$

5.2 Wireless server model

In the "wireless server" model, the server is embedded on an additional wireless device associated with the AP, as depicted in Figure 2c. In this section, we

extend the previous model to deal with real-world implementation constraints. The paradigm shifts from the evaluation of the aggregation levels of the UL probe traffic to the aggregation levels of the DL probe traffic forwarded by the AP to the server. Two Markov chains are proposed for this model. In the following, we only present the Markov chain based on aggregated cross traffic.

We consider the Markov chain defined as $(X_n, Y_n, Z_n, S_n)_{n\geq 0}$. The process X_n defines the number of probe traffic sub-frames contained in the AP queue at the moment of the n^{th} frame transmission departure. The set of possible states is $\{0, .., AMPDU_{AP}\}$. The random process Y_n describes the number of cross traffic sub-frames in the AP queue. The possible states are $\{0, .., AMPDU_{AP}\}$. The process Z_n describes the number of packets at the probe traffic node. Its possible states are $\{0, .., AMPDU_P\}$. Let S_n be the n^{th} transmission. It takes three possible values $\{APC, APP, SP\}$. Let us denote by APP (Access Point Probe) the DL transmission of probe traffic from the AP. Let APC (Access Point Cross) be the cross traffic transmission from the AP to the cross server, and let SP (Station Probe) be the UL transmission from the probe traffic node.

Transition probabilities We now derive the transition probability denoted $P_{(i,j,k,u)(l,m,q,v)}$ which represents the probability to go from state $(X_n, Y_n, Z_n, S_n) = (i, j, k, u)$ to state $(X_{n+1}, Y_{n+1}, Z_{n+1}, S_{n+1}) = (l, m, q, v)$.

Conversely to the ideal server model, not all transitions are possible. So, the next step consists in deciding when a transition from one state to another is allowed and computing its probability. It is worth noting that impossible transitions have zero probability and that the associated transition probability is computed by assuming that all concerned nodes are equally likely to access the channel. We classify the state transitions into three classes. For ease of presentation, we only present the first class in the following. The two other classes are presented in C.

Class I:Transition from state APP Figure 4 represents the possible transition probabilities and their computations. It shows a set of possible events between two successive transmissions when the current transmission is APP (an aggregated frame containing probe traffic packets is transmitted by the AP). In this example, at time $t = 0$, we start from state (i, j, k, APP) (i.e (3,2,2,APP) in Figure 4). The AP sends the i corresponding probe frames currently in its buffer ($i = 3$ in this example), aggregating them in a single A-MPDU using its current Modulation Coding Scheme (MCS) index. Its transmission duration is denoted $T_{AP}(i)$ which is given in A. It is important to note that the AP has a unique buffer that contains at once the probe and the cross traffics, but in order to clarify the explanation of the model, we distinguish between them in Figure 4.

During this transmission, the probe node's buffer receives $\lfloor \frac{T_{AP}(i)}{d_p} \rfloor$ packets and the AP buffer receives $\lfloor \frac{T_{AP}(i)}{d_c} \rfloor$ cross packets. As a result, at the end of the APP state, the probe traffic node buffer and the AP will contain respectively,

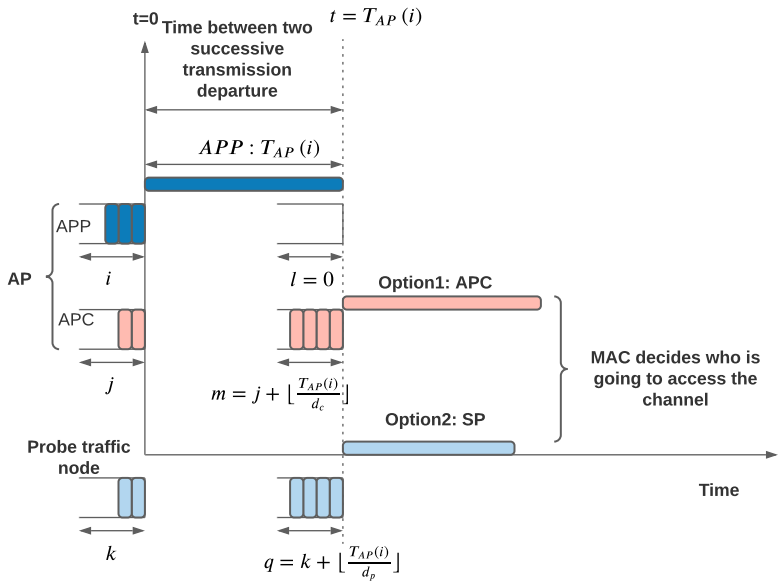


Figure 4: Possible transitions from state APP

$k + \lfloor \frac{T_{AP}(i)}{d_p} \rfloor$ of probe packets and $j + \lfloor \frac{T_{AP}(i)}{d_c} \rfloor$ of cross traffic packets ((4,4) in the Figure 4).

At the end of this transmission, the buffer of the AP does not have probe frames to send ($X_{n+1} = 0$ almost surely), and another APP transmission is impossible ($S_{n+1} \neq APP$ almost surely). So from this state, only two transitions are allowed: to APC or SP with $X_{n+1} = l = 0$. It can occur only if $Y_{n+1} = m > 0$ and $Z_{n+1} = q > 0$ respectively.

We derive the non-null transition probabilities as follows. If the AP gains access to the channel, it will send the cross traffic currently in its buffer. For $m > 0$, the next transmission will be APC with probability:

$$P_{(i,j,k,APP)(0,m,q,APC)} = P\Big(S_{n+1} = APC | S_n = APP, Z_{n+1} = q, Y_{n+1} = m > 0\Big) \cdot \mathbb{1}_{k+\lfloor \frac{T_{AP}(i)}{d_p} \rfloor = q} \cdot \mathbb{1}_{j+\lfloor \frac{T_{AP}(i)}{d_c} \rfloor = m} \tag{6}$$

where $\mathbb{1}_{condition}$ is the indicator function that equals to 1 if *condition* is true and 0 otherwise. $P\Big(S_{n+1} = APC | S_n = APP, Z_{n+1} = q, Y_{n+1} = m > 0\Big)$ denotes the probability that the event APC will occur given that the event APP has already occurred. In the interest of brevity, we adjourn the computation of such probabilities to D.

Now, if the probe station gains access, the next event will be SP, and the transition probability from APP to SP for $q > 0$ is given by:

$$P_{(i,j,k,APP)(l,m,q,SP)} = P\Big(S_{n+1}= SP|S_n = APP, Y_{n+1} = m, Z_{n+1} = q > 0\Big) \cdot \mathbb{1}_{k+\lfloor \frac{T_{AP}(i)}{d_p} \rfloor = q} \cdot \mathbb{1}_{j+\lfloor \frac{T_{AP}(i)}{d_c} \rfloor = m} \tag{7}$$

5.3 Stationary probability

So far, we have evaluated all the transition probabilities for the two models. The last step consists in deriving the stationary probability and computing the frame aggregation levels. Let us recall that a Markov chain is irreducible if and only if every state can be reached by any other state through one or several transitions. Since our Markov chain is irreducible and has a finite number of states, it exists a unique stationary distribution [28]. We solve this Markov chain through a numerical method. We compute the stationary probabilities. The vector containing the corresponding values is denoted π. The mean probe traffic aggregation levels for the first and second models denoted respectively $MeanAgg_{Ideal\ server}$ and $MeanAgg_{Wireless\ server}$ are then given by:

$$MeanAgg_{Ideal\ server} = \sum_{n=1}^{AMPDU_P} n \cdot \pi_n$$

.

$$MeanAgg_{Wireless\ server} = \sum_{n=1}^{AMPDU_{AP}} n \cdot \pi_n$$

.

6 Description of FAM

We now describe our scheme, FAM, that allows the node conducting the measurement to estimate the BTF of the wireless channel and to infer the type of the cross traffic. This method relies on the three following steps.

- Measure the mean aggregation level for different probe flows (Section 6.1)
- Detect the use of frame aggregation in the cross traffic (Section 6.2)
- Estimate the wireless channel load (Section 6.3)

All the algorithms are implemented on the server except the client procedure (Algorithm 1).

6.1 Mean aggregation level

The measurement of the aggregation level is described in algorithms 1 and 2. Algorithm 1 runs on the probe node. It starts by sending successive batches of probe packets. For each batch, the gap interval between packets, d_p, is increased from $d_{p_{min}}$ to $d_{p_{max}}$. In order to evaluate the mean aggregation level accurately while keeping the batch size as small a possible, the number of packets in a batch np is not fixed. This number is determined using the Central Limit Theorem (CLT) detailed in the server procedure (Algorithm 2). Upon receipt of this batch, the server responds with a packet that indicates to the probe node to stop sending packets. It is the *convergence* variable in the algorithm.

Algorithm 1 Client Procedure

Input: K_{max} : Maximum A-MPDU Length, np : size of a batch of probes

1: $d_{p_{min}} = \frac{f(K_{max})}{K_{max}}$
2: $d_p = d_{p_{min}}$
3: **repeat**
4: **repeat**
5: client.send (d_p, np) ▷ Send np packets with interval d_p
6: client.receive(convergence)
7: **until** convergence is True
8: client.receive($meanAgg$)
9: $d_p = d_p$+increment
10: **until** $meanAgg > 2$
11: client.send(0, 1) ▷ Send 1 packet to inform that the campaign is finished.

Algorithm 2 Server Procedure

Input: E :acceptable standard error of the mean, Z : Value of the distribution function

MeanAgg=$\emptyset$, D=$\emptyset$
while campaign in progress for this client **do**
  $n = 0$, convergence = False, $meanAgg = 0$
  while !convergence **do**
    server.receive(1) ▷ Reception of the first packet that contains the parameters (d_p, np)
    server.receive($np - 1$) and store $agg_{i+n},\ \forall i \in \{1, np\}$
    $n = n + np$
    $meanAgg = \frac{\sum_{j=1}^{n} agg_j}{n}$
    $S^2 = \frac{1}{n-1} \sum_{i=1}^{n} (agg_i - meanAgg)^2$
    $convergence = \left(n \geq \frac{Z^2 \times S^2}{E^2}\right)$
    server.send(convergence)
  end while
  Add $meanAgg$ to vector MeanAgg
  Add d_p to vector D
  server.send($meanAgg$)
end while

Output: (MeanAgg, D) ▷ Returns two vectors: the "MeanAgg" and "D"

Algorithm 2 runs on the server. It receives and processes the probe packets from the probe node. During a given batch, the mean aggregation is computed on the fly at the reception of probe packets. We use the CLT to evaluate the accuracy of the mean aggregation level. When the error is lower than the expected error E, the server sends the *convergence* notification to the probe node (that stops its batch). The use of the CLT allows the application to appropriately measure the mean aggregation level with a minimal amount of time and bandwidth cost.

6.2 Cross traffic nature

Our reasoning to assess the cross traffic nature is based on the idea that, if the cross traffic does not aggregate, its channel access time becomes constant when the channel load increases. This time is defined here as the time the cross traffic uses the channel between two successive probe traffic accesses. It corresponds to $g+g$ in Figure 3. The cross traffic access time depends only on the successive number of times it accesses the channel and the time to transmit a single frame. When the two buffers (cross and probe) are non-empty, it becomes independent of the number of frames/packets in these buffers and thus on the probe and traffic loads. On the contrary, when the cross traffic aggregates its frames, this time is increasing with the load as A-MPDU contains more aggregated frames.

We use this observation to detect the nature of the cross traffic. The mean

cross traffic access time is denoted T_C. If this time is constant with d_p and with the load, we can express the mean aggregation level for the probe traffic, $meanAgg$, through a fixed point equation.

$$meanAgg = min\left(K_{max}, \frac{f(meanAgg)}{d_p} + \frac{T_C}{d_p}\right) \quad (8)$$

The rational of this equation is that the mean number of aggregated frames ($meanAgg$ on the left-hand side of the equation) is equal to the number of frames received at the buffer since the last probe transmission. In Figure 3, we would get $X_{n+1} = \frac{f(X_n)}{d_p} + \frac{g+g}{d_p}$. Substituting (X_n, X_{n+1}) by the mean aggregation ($X_n = X_{n+1} = meanAgg$) and assuming that the cross traffic access time is constant, $(g + g)$ becomes T_C for this example), we obtain the Equation 8.

This approach is described in Algorithm 3. It takes as input the mean aggregation level for the probe and computes T_C, according to equation 8, for each probe packet gap d_p. It keeps only values of T_C for which $\frac{f(meanAgg)}{d_p} + \frac{T_C}{d_p}$ is less than K_{max}. Then, it returns the percentage of variation between the maximum and minimum of these values. In Algorithm 6, this percentage is compared to a given threshold to determine if T_C can be considered as constant (cross traffic does not aggregate) or not (cross traffic aggregates).

Algorithm 3 Percentage Increase Algorithm

Input: MeanAgg=$(meanAgg(d_{p_{min}}), ..., meanAgg(d_{p_{max}}))$,
D=$(d_{p_{min}}, ..., d_{p_{max}})$
1: $TC\_VECTOR = \emptyset$
2: **for** all $d_p \in D$ **do**
3: Compute T_C:
4: $T_C(d_p) = d_p * meanAgg(d_p) - f(meanAgg(d_p))$
5: **if** $\frac{f(meanAgg)}{d_p} + \frac{T_C}{d_p} < K_{max}$ **then**
6: Add $T_C(d_p)$ to vector TC_VECTOR
7: **end if**
8: **end for**
9: $\%increase \leftarrow \frac{max(TC\_VECTOR) - min(TC\_VECTOR)}{min(TC\_VECTOR)} * 100$
Output: $\%increase$

6.3 BTF estimation

Two algorithms are proposed to estimate the BTF. They are both based on the comparison between the mean aggregation levels given by our Markov chains and the measures made for different probe traffic batches.

Algorithm 4 computes the mean error as the sum of the difference between the model and the measures. It returns the BTF that minimizes this error. The considered levels of BTF are a parameter of the algorithm. It returns a BTF

Algorithm 4 BTF Error Based Method Algorithm

Input: MeanAgg=$(meanAgg(d_{p_{min}}), ..., meanAgg(d_{p_{max}}))$,
D=$(d_{p_{min}}, ..., d_{p_{max}})$
1: **for each** $case \in \{w/A, wo/A\}$ **do**
2: **for** $bt \in$ levels of BTF **do**

$$Error_{(bt,case)} = \frac{1}{Size(MeanAgg(.))} \sum_{d_p=d_{p_{min}}}^{d_{p_{max}}} \mid model_{(case,d_p,bt)} - meanAgg(d_p) \mid$$

3: **end for**
Output: $(\text{argmin}_{bt}\, Error_{(bt,w/A)}, \text{argmin}_{bt}\, Error_{(bt,wo/A)})$
4: **end for**

for the model with aggregation (w/A) and a BTF for the model without frame aggregation (wo/A).

Algorithm 5 considers a notion of score to infer the BTF. For each probe packet gap d_p and a given model, the BTF that minimizes the error scores one. For each model, the final BTF will be the one that maximizes this score.

Algorithm 5 BTF Score Based Method Algorithm

Input: MeanAgg=$(meanAgg(d_{p_{min}}), ..., meanAgg(d_{p_{max}}))$,
D=$(d_{p_{min}}, ..., d_{p_{max}})$
1: **for** $d_p \in D$ **do**
2: **for each** $case \in \{w/A, wo/A\}$ **do**
3: **for** $bt \in$ all levels of BTF **do**
4: $Error(case, d_p, bt) = \frac{1}{Size(meanAgg(.))} \mid model(case, d_p, bt) - meanAgg(d_p) \mid$
5: **end for**
6: **end for**
7: $Score(\text{argmin}_{bt,case}\, Error(case, d_p, bt), case) += 1$
8: **end for**
Output: $(\text{argmax}_{bt}\, Score(bt, w/A), \text{argmax}_{bt}\, Score(bt, wo/A))$

Algorithm 6 gives the final output of our method. First, it evaluates if the BTF is less or equals to 0.25. We have observed that, for such a low load, it is difficult to distinguish precisely the nature of the traffic since the cross traffic tends to have very small mean aggregation levels. In this case, no precision on the aggregation is given. If the BTF is greater than 0.25 at least for one of the two models, it computes the percentage of variation of T_C with Algorithm 3 and compares it to a threshold T. It determines if the cross traffic access time can be considered as constant and consequently if the cross traffic aggregates or not. For both cases, the corresponding BTF is returned with the cross traffic nature.

Algorithm 6 BTF Estimation Algorithm

Input: T:Threshold, E: acceptable standard error of the mean

(MeanAgg, D) = Server Procedure(E, Z)
$PI = PercentageIncrease(MeanAgg, D)$
$BTF_{w/A}, BTF_{wo/A}$ = Error Computation Procedure(MeanAgg,D)
$BTF1_{w/A}, BTF1_{wo/A}$ = Score Computation Procedure(MeanAgg,D)
if ($BTF_{w/A} \leq 0.25$ *or* $BTF1_{w/A} \leq 0.25$) *and* ($BTF_{wo/A} \leq 0.25$ *or* $BTF1_{wo/A} \leq 0.25$) **then**
    return $\{wo/A, BTF \leq 0.25\}$
else if $0 < PI < T$ **then**
    return $\{wo/A, BTF > 0.25\}$
else
    return $\{w/A, BTF_{w/A}\}$
end if

7 Numerical results

In this section, we start by evaluating the accuracy of the proposed models and FAM under several scenarios with different network parameters, such as the topology and its size, different IEEE 802.11 amendments, and different traffic patterns. In this regard, we compare the aggregation levels given by the models with those delivered by the discrete-event network simulator 3 (ns-3 version 3.30) [29]. We also compare the models' aggregation levels with the measurements made during a test-bed experiment and a real-world trace-driven evaluation in order to get realistic scenarios that capture the complexity of the whole network stack. Then, we study the difference between the proposed Markovian models. At last, we explore the detection of the aggregation level at the application layer.

The parameters of our method are as follow. We compute the models for six cross traffic loads/BTF levels: 0, 0.125, 0.25, 0.375, 0.5, 0.625 ranging from low to high levels of BTF. The cross traffic is a flow of UDP packets generated at a specified rate to create six levels of load. In algorithm 2, the FAM method uses a 95% confidence interval ($Z = 1.96$) and an error $E = 0.05$.

The CLT results indicated that the relevant number of packets to send increases proportionally to the probe rate.
The analysis of the traces measured during both the simulations and experiments described below show that our method produces at most 5 MB of data if we use a packet size of 1024 Bytes down to 0.5 MB if we use a packet size of 100 Bytes. For a point of comparison, Iperf3 produces over 11 MB of data in these settings.

In algorithm 6, the threshold T is chosen according to an empirical method. Our simulations and experiments have shown that a value of $T = 200\%$ offers good performances. For instance, we take different values for the parameter T that depend on the device's transmission rate. These values are known by the method FAM.

7.1 Ideal server model based on aggregated cross traffic validation

We start by examining the accuracy of ideal server model and FAM using the Markov chain where frame aggregation mechanism is enabled for both probe and cross traffics under the IEEE 802.11n amendment [27]. We compare it to ns-3 simulations and a controlled lab experiment.

7.1.1 Experimental setup

The analytical results were compared and validated using an experimental test-bed. This experiment (illustrated in Figure 5) was conducted in a controlled lab environment (a helical room) where there is no interference or channel fading [30].

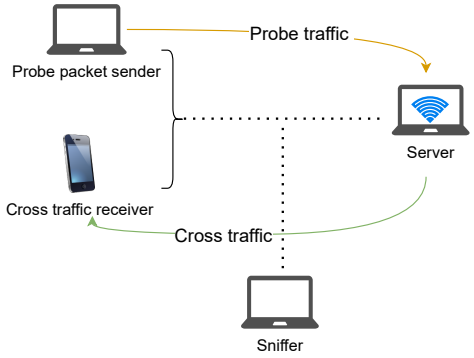


Figure 5: Experimental test-bed

The general setting is as follows: we used a Linux laptop to execute the probe sender application. Another computer, configured as an IEEE 802.11n AP, was used as a server for the probe and client for the cross traffic. The physical transmission rate was set to 144.4 Mbps (i.e., MCS index 15 in 802.11n). An Android phone was deployed, acting as the cross traffic receiver application. Also, we configured a computer (Sniffer) with a specific Wireless Network Interface Controller (WNIC) that supports the "survey dump" feature of the "iw" command which is a Linux utility that shows the survey information of all the available channels including the "channel busy time" and "channel active time", thereby we measure the ground truth value of the BTF as follows. $BTF = \frac{channel\ busy\ time}{channel\ active\ time}$. It is worth noting that not all drivers support this feature. This computer is also configured in monitor mode to capture the aggregated frames by using a dedicated capture software. The aggregation level is computed according to the A-MPDU reference number in the radiotap header (additional information added by the wireless adapter or its driver).

All the nodes operate on channel 1 in the 2.4GHz band with 20MHz bandwidth.

7.1.2 Simulation configuration

In our Markov chain based models, the cross traffic is sent by a single queue. In the simulated scenario, we model cross traffic sent to four concurrent nodes. The network topology is hence composed of an AP and five nodes, illustrated in Figure 6.

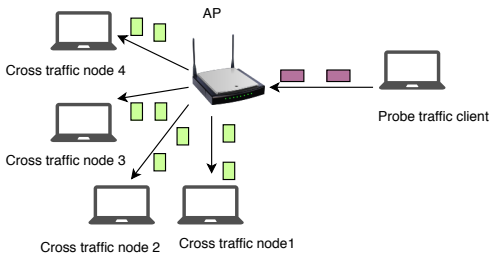


Figure 6: ns-3 simulation for the ideal server model based on an aggregated cross traffic

7.1.3 Evaluation

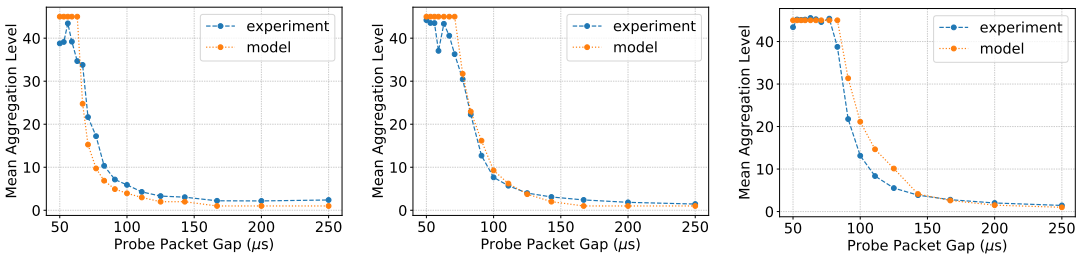
We now analyze the performance of the first Markov chain for the ideal server model. Figure 7 shows the mean aggregation level for the probe traffic as a function of the probe traffic interval for the analytical model (Orange curve) and experiments (Blue curve). In these experiments, aggregation is enabled for the cross traffic.

Experiments are therefore compared to the model based on aggregated cross traffic outcomes. The probe packet gap gradually increases from 50μs to 250μs. Due to space limitations, we show only the results of 0, 0.25, and 0.5 BTF, since the other cases present the same accuracy.

We observe that our model is able to capture with a reasonable accuracy the experimental mean aggregation level of the probe traffic for all the levels of the cross traffic. Not surprisingly, for a very high level of probe traffic (small probe packet gaps), the mean aggregation level reaches the maximum. Then, it decreases until reaching 1.

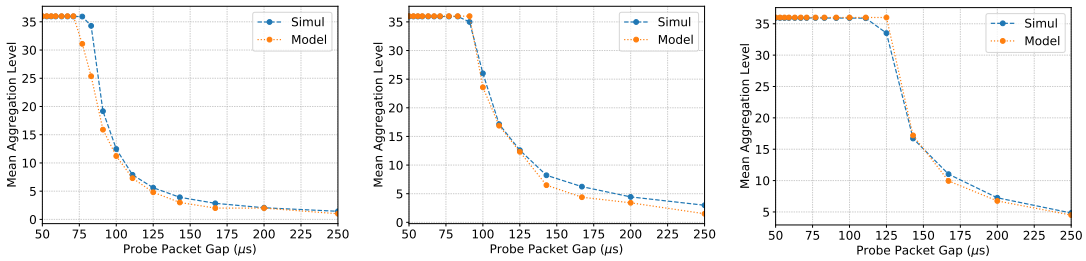
Similarly, in Figure 8, we compare the mean aggregation level obtained with ns-3 with the model based on aggregated cross traffic. The corresponding results show that the mean aggregation levels derived from simulations are always close to our model, and all the curves show similar patterns.

Overall, these results demonstrate that despite the complexity brought by the network stack layers (beacon frames, congestion, ARQ, random backoff, etc.) and the different number of stations deployed in the topology, our approach captures the mean aggregation level with a reasonable level of precision in the considered scenarios.



(a) BTF=0 (b) BTF = 0.25 (c) BTF = 0.5

Figure 7: Mean aggregation level: ideal server model versus experiments



(a) BTF=0.125 (b) BTF=0.375 (c) BTF=0.625

Figure 8: Mean aggregation level: ideal server model versus ns-3 simulations

Table 2: BTF and cross traffic nature estimations for test-bed experiments

Ground Truth BTF	Estimated BTF and cross traffic nature					
	Cross traffic aggregates				Cross traffic does not aggregate	
	≤ 0.25	0.375	0.5	0.625	≤ 0.25	> 0.25
0.125	Exp/A				Exp/A	
0.25	Exp/A				Exp/A	
0.375	A	Exp				
0.5		A	Exp			
0.625			A	Exp		

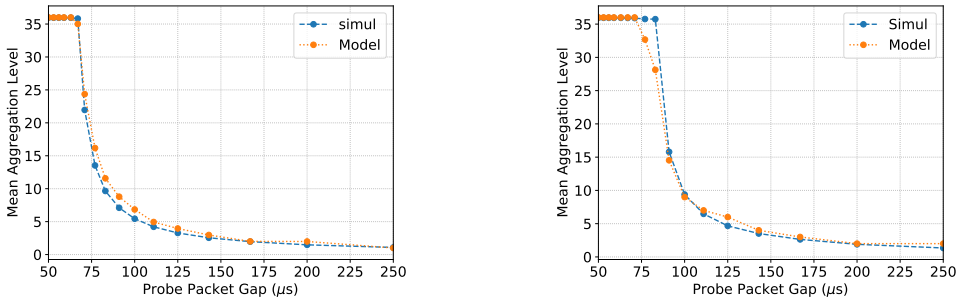
We now evaluate the capacity/ability of the proposed method FAM to infer BTF and cross traffic nature. Table 2 presents the estimated BTF and cross traffic nature for the experiments. The 'ground truth' column gives the real value of the BTF. In the table, 'Exp' indicates the BTF and cross traffic nature that were set in the experiments, and 'A' presents the value returned by our algorithm FAM. FAM finds the good result when the two letters ('Exp'/'A') are in the same box. As FAM does not return the cross traffic nature when the load is lower than 0.25, we set 'Exp' and 'A' in the two corresponding boxes (aggregated and non-aggregated cross traffic) in this case.

Results show that the method finds the good results for the two lowest BTFs and slightly underestimates the BTF for the last three cases (O.375, 0.5 and 0.625). This error is only of 0.125 (approximately 10%) and FAM returns the precise nature of the cross traffic for all cases.

Table 3 lists the estimated BTF and cross traffic nature returned by FAM for the ns-3 simulations. In this Table, the letter 'S' referred to the simulated parameters. FAM provides the good BTF except for the fourth case: BTF=0.5 with aggregation. For all the cases, it is able to infer that the cross traffic is aggregated (when BTF$>$ 0.25). We note that FAM seeks to infer the BTF based on six levels of loads ranging from 0 to 0.625 with a resolution of 0.125. It tends to classify the APs into three categories according to the network load: low, medium, and high network loads to detect the APs that are less loaded. In table 3, we ran the ns-3 simulations with two new network loads: 0.15 and 0.4. For BTF=0.15, the method returns that the BTF is less or equal to 0.25, and the cross traffic is aggregated, which is true. For BTF = 0.4, the method returns that the BTF = 0.375 since it is the nearest value to 0.4.

Table 3: BTF and cross traffic nature estimations for ns-3 simulations

Ground Truth BTF	Estimated BTF and cross traffic nature					
	Cross traffic aggregates				Cross traffic does not aggregate	
	≤ 0.25	0.375	0.5	0.625	≤ 0.25	> 0.25
0	S/A				S/A	
0.15	S/A				S/A	
0.125	S/A				S/A	
0.25	S/A				S/A	
0.375		S/A				
0.4		S/A				
0.5		A	S			
0.625				S/A		



(a) BTF = 0

(b) BTF = 0.125

Figure 9: Mean aggregation level versus BTF, cross traffic without frame aggregation

7.2 Ideal server model based on non-aggregated cross traffic

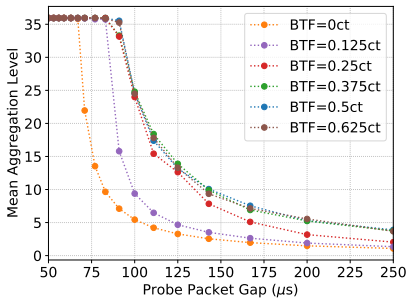
We examine the proposed approach's accuracy when the frame aggregation mechanism is disabled for the cross traffic. The considered scenario consists of two co-located IEEE 802.11n and IEEE 802.11g WLANs on the 2.4 GHz band. The 802.11n WLAN is composed of an AP and a node that sends the probe traffic, while the 802.11g is composed of an AP and a node that receives the non-aggregated cross traffic.

We compare the aggregation given by the models detailed in Section 5 to the values obtained by the ns-3 simulations in Figures 9a and 9b. It appears that the model is able to estimate the mean aggregation level with a reasonable accuracy for the two BTFs. The difference that can be observed is due to the fact that beacons are not taken into account in the model, that is why it underestimates the aggregation level for some probe packet gaps. We note that our models take into account the time to send block ACK requests while computing the frame transmission duration.

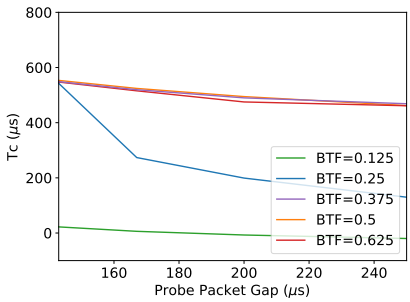
Figure 10a shows the probe mean aggregation level for each of the six BTFs. Based on these results, we can see that varying the level of cross traffic barely affects the probe aggregation level when the $BTF > 0.25$. As the cross traffic does not aggregate, its access time does not depend on its buffer state. Consequently, the aggregation level of the probe traffic becomes insensitive to the level of congestion of the cross traffic.

It is also illustrated in Figure 10b that shows the mean cross traffic access time, T_C (computed according to equation 8), as a function of the probe packet gap. It appears clearly that T_C is constant with d_p and with the load when the $BTF > 0.25$.

Table 4 lists the estimated BTF and cross traffic nature returned by FAM for each of the BTF levels. For all the cases, FAM performs extremely well and provides the expected level of the channel load and the precise nature of the



(a) Mean Aggregation level for all BTF simulation



(b) Tc versus Probe packet gap

Figure 10: Mean aggregation level and T_C versus BTF, cross traffic without aggregation

Table 4: BTF and cross traffic nature estimations for ns-3 simulations, non-aggregated cross traffic

Ground Truth BTF	Estimated BTF and cross traffic nature					
	cross traffic aggregates				cross traffic does not aggregate	
	≤ 0.25	0.375	0.5	0.625	≤ 0.25	> 0.25
0.125	S/A				S/A	
0.25	S/A				S/A	
0.375						S/A
0.5						S/A
0.625						S/A

cross traffic.

7.3 Additional simulation results for the ideal server model

For the sake of completeness, we also ran our model and our method FAM using a mix of an aggregated and non-aggregated cross traffic scenarios (based on 802.11n and 802.11g cross traffic) and another scenario based on the recent IEEE 802.11ax standard [31].

7.3.1 Mixed cross traffic

We investigate the BTF of a mixed cross traffic. We present two scenarios. The network topology is depicted in Figure 11. We simulate a scenario where the mixed cross traffic is composed of 80% of aggregated traffic and 20% of non-aggregated traffic (Scenario 1) and inversely (Scenario 2).

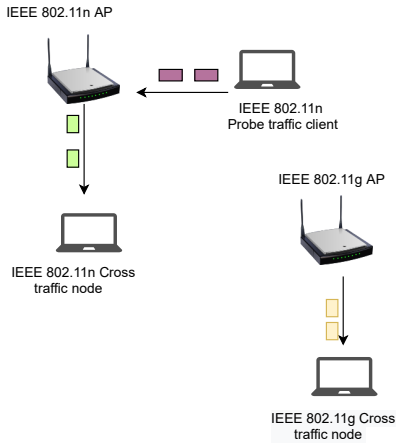


Figure 11: ns-3 simulation for the mixed cross traffic

Tables 5 and 6 provide the predictions made by scheme FAM. Table 5 shows that the method is able to predict the good results for the two lowest BTFs, 0.125 and 0.25. Since the cross traffic is composed of 80% of aggregated traffic, the method tends to predict reasonably well the nature of the cross traffic (Aggregates) for the highest BTFs 0.375, 0.5, and 0.625; however, the predicted values were slightly overestimated.

Table 6 reveals that FAM returned the same results as the cases of 100% non-aggregated cross traffic, except for the third case: $BTF = 0.375$. Note that we have $0.375 * 0.8 = 0.3$ of non-aggregated traffic in this case. It is close to the case with a BTF of 0.25 of cross traffic with non-aggregation for which it has been shown that the access time T_C varies significantly (Figure 10b).

Table 5: BTF and cross traffic nature estimations for ns-3 simulations with mixed cross traffic, 80% of aggregated traffic and 20% of non-aggregated traffic

Ground Truth BTF	Estimated BTF and cross traffic nature					
	cross traffic aggregates				cross traffic does not aggregate	
	≤ 0.25	0.375	0.5	0.625	≤ 0.25	> 0.25
0.125	S/A				S/A	
0.25	S/A				S/A	
0.375		S	A			
0.5			S	A		
0.625				S/A		

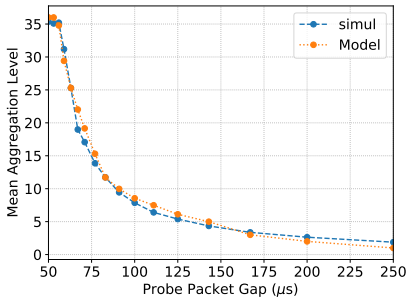
Table 6: BTF and cross traffic nature estimations for ns-3 simulations with mixed cross traffic, 20% of aggregated traffic and 80% of non-aggregated traffic

Ground Truth BTF	Estimated BTF and cross traffic nature					
	cross traffic aggregates				cross traffic does not aggregate	
	≤ 0.25	0.375	0.5	0.625	≤ 0.25	> 0.25
0.125	S/A				S/A	
0.25	S/A				S/A	
0.375			A			S
0.5						S/A
0.625						S/A

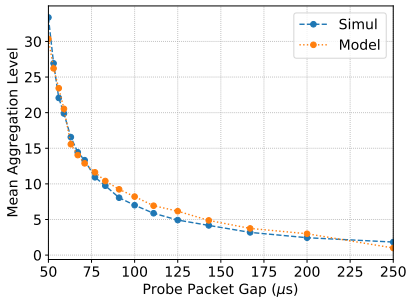
Note that in order to investigate the robustness of our approach, we explored two other ns-3 scenarios where the mixed cross traffic is composed of 60% of aggregated traffic and 40% of non-aggregated traffic (first scenario) and inversely (Second scenario). The corresponding results are not presented in this paper. However, they show that FAM's accuracy is similar to the two previous mixed scenarios.

7.3.2 IEEE 802.11ax simulation results

In our last scenario, we study the model's and methodology's accuracy when we change the underlying Wi-Fi network from 802.11n to 802.11ax in the 2.4 GHz band. We used the current implementation of the IEEE 802.11ax of ns-3. All the nodes (AP and stations) operate at two data rates, 286.8$Mbps$ (High efficiency (He)MCS11, spatial streams=2, 20MHz with 1024QAM, 0.8GI) and 2402$Mbps$ (HeMCS11, spatial streams=2, 160MHz with 1024QAM, 0.8GI). The



(a) $BTF = 0.25$, Data rate = 286.8 Mbps



(b) $BTF = 0.25$, Data rate = 2404 Mbps

Figure 12: Mean aggregation level: ideal server model versus ns-3 simulations. The cross traffic is aggregated.

network topology is composed of an AP and two nodes. A probe node sends the probe traffic, and another node receives the aggregated cross traffic from the AP. We run our FAM method while considering an empirical threshold T equal to 100 for transmission rate= $286.8Mbps$ and T equal to 40 for transmission rate=$2402Mbps$. This threshold is adapted according to the transmission rate.

Figure 12 shows that the mean aggregation levels obtained with simulations fit the ones from the model for both the low and the high data rates.

The results of FAM are shown in Table 7. It can be seen that FAM predicts reasonably well the BTF values and the cross traffic nature with typically 10% of error for the two cases.

7.3.3 Trace driven evaluation

In order to validate the accuracy of the ideal server model under real-environment conditions, we conducted a trace driven ns-3 scenario. For collecting the trace, we ran a sniffer capture to record the traffic of the train station "Part Dieu" in the french city, Lyon. This capture is long enough (200s) to collect a trace that includes a variety of channel conditions, mobility, multiple APs, multiple user stations, WiFi, and non-WiFi interference. This real-word captured dataset is then injected in ns-3 as the cross traffic. By adjusting a time factor that will be multiplied by the cross traffic inter-arrivals, we obtained four different network loads: 0.11, 0.328, 0.459, and 0.729.

Table 7: BTF and cross traffic nature estimations for ns-3 simulations using 802.11ax standard

(a) Transmission rate = 286.8 Mbps

	Estimated BTF and cross traffic nature					
Ground Truth BTF	cross traffic aggregates				cross traffic does not aggregate	
	≤ 0.25	0.375	0.5	0.625	≤ 0.25	> 0.25
0	S/A				S/A	
0.125	S/A				S/A	
0.25	S/A				S/A	
0.375		S/A				
0.5		A	S			
0.625				S/A		

(b) Transmission rate = 2402 Mbps

	Estimated BTF and cross traffic nature					
Ground Truth BTF	cross traffic aggregates				cross traffic does not aggregate	
	≤ 0.25	0.375	0.5	0.625	≤ 0.25	> 0.25
0	S/A				S/A	
0.125	S/A				S/A	
0.25	S/A				S/A	
0.375		S	A			
0.5			S/A			
0.625				S/A		

Table 8: BTF and cross traffic nature estimations for ns-3 simulations where the cross traffic is aggregated, Ideal server model

Ground Truth BTF	Estimated BTF and cross traffic nature					
	Cross traffic aggregates				Cross traffic does not aggregate	
	≤ 0.25	0.375	0.5	0.625	≤ 0.25	> 0.25
0.11	S/A				S/A	
0.328		S/A				
0.459		A	S			
0.729				S/A		

The results of FAM are shown in Table 8. We can observe that FAM predicts well the BTF values and the cross traffic nature for the cases 0.11, 0.328 and 0.729 cases and slightly underestimates the BTF for the case 0.459.

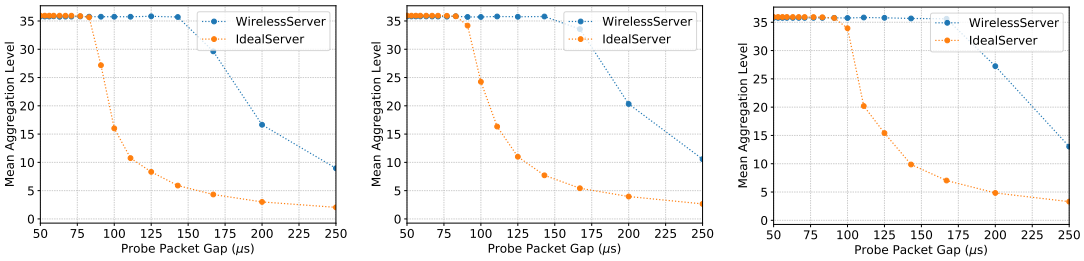
7.4 Validation of wireless server model

In order to assess the effectiveness of wireless server model and FAM, we explored all the aforementioned scenarios used to evaluate the accuracy of ideal server model. The corresponding results are not presented in this paper owing to lack of space. However, they show that the wireless server model has the same accuracy as the ideal server model. In the following, we evaluate wireless server model's and FAM's accuracy under a new scenario that considers an exponential aggregated cross traffic.

Table 9: BTF and cross traffic nature estimations for ns3 simulations where the cross traffic is aggregated and exponential under wireless server model

Ground Truth BTF	Estimated BTF and cross traffic nature					
	Cross traffic aggregates				Cross traffic does not aggregate	
	≤ 0.25	0.375	0.5	0.625	≤ 0.25	> 0.25
0	S/A				S/A	
0.125	S/A				S/A	
0.25	S/A				S/A	
0.375		S/A				
0.5			S	A		
0.625				S/A		

In the Markov chain models, the cross traffic is assumed to be deterministic. Table 9 shows the results returned by FAM when the cross traffic is exponential. It can be seen that FAM predicts reasonably well the BTF values and the



(a) BTF=0.25 (b) BTF=0.375 (c) BTF=0.5

Figure 13: Mean aggregation level: ideal server versus Wireless server

cross traffic nature for the 0, 0.125, 0.25, 0.375 and 0.625 cases and slightly overestimates the BTF for the 0.5 case.

7.5 Comparison between the models

We now compare the two proposed models according to the aggregation levels. Figure 13 provides the mean probe aggregation levels when frame aggregation is enabled for the cross traffic for the two models. It can be seen that the UL aggregation levels obtained by the ideal server model are lower than the DL aggregation levels returned by the wireless server model. Indeed, the behavior of the aggregation levels is more complex since the DL probe traffic depends on the competition for medium access with the UL probe traffic and the cross traffic. In particular, when the cross traffic increases, the DL probe traffic has to wait longer. More packets may hence accumulate in the queue of the AP and be aggregated when the next DL probe transmission occurs.

7.6 Detection of the aggregation level at the application layer

As highlighted in Section 7.1.1, in order to get the aggregation level ground truth in the experiment, we used the packet A-MPDU reference captured by the sniffer. Since the final targeted application cannot use this method, our technique will rely on a threshold-based method that computes the inter-arrival time between two packets at the application layer. If the gap time between two consecutive received packets is less than a given threshold, the corresponding packets are deemed to be aggregated. To assess the effectiveness of this approach, we conducted further evaluation experiments. Our experimental environment was set up as follows. An application was executed on an Android smartphone to send a set of probe packets to the server. The server code was written in C and executed on a laptop running Ubuntu 18.04. The client, as well as the server, were connected to a 802.11 Linksys LAPAC1750 AP. We ran this experiment at noon in a residence with all the real wireless interference.

In Figure 14, we compare the mean aggregation levels computed by the server with those captured by the sniffer. Several experiments have been performed with equivalent results, the figure shows one of them. We tested several thresholds. For these experiments, a value of 250 μs gives inferred aggregation levels that match well with the ones observed by the sniffer. Note that this threshold depends on the system used. Since our definitive application would rely on dedicated servers depending on the probing node location, the threshold values for these servers are known by our method.

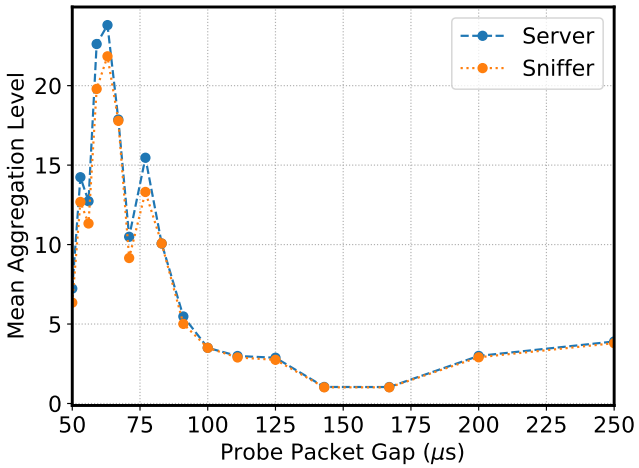


Figure 14: The mean aggregation levels captured by the server versus the sniffer

8 Discussion

The method proposed in this paper is a proof of concept that shows that we can infer the network load. Our method can be used if the server is closed to the AP on the wired network, or when two users embedding the application are connected to the same AP. The latter assumes that the AP does not filter communication between associated stations, which is not always the case. Our solution is thus not applicable in all contexts. In the other cases, stations should use a classical method to estimate their available bandwidth, for instance, that tests the maximum amount of data per second that can be sent to one (or several servers) on the Internet. A survey of the existing tools can be found in [32, 33, 34]. These techniques are less adapted to our application as they evaluate the maximum throughput that a given station can obtain and not the Wi-Fi load. It can then be more difficult to transpose the obtained throughput to other stations. Nevertheless, they offer a good alternative for cases where our method is not yet applicable.

9 Conclusion and future work

In this paper, we have developed a new proof of concept method, FAM, that allows a vanilla device, typically a smartphone, to infer the Wi-Fi network load based on the observed frame aggregation level in the user space, and predicts if the current traffic aggregates its frames or not. This is meant as a building block for a crowd-sourcing application, in which participant devices measure and share the load of their surrounding networks. They would build a collective knowledge so that when a device arrives in some area and wants to connect to a Wi-Fi network, it could choose the less loaded AP.

We studied a variety of scenarios to assess the effectiveness of our method by comparing its outcomes with those delivered by the ns-3 simulator, test-bed experiments, and an ns-3 trace driven scenario based on real-world data. We considered several network topologies, different IEEE 802.11 amendments, several cross traffic patterns, various levels of the channel load as well as scenarios where the cross traffic is a mix of aggregated and non-aggregated traffics.

Overall, we find that FAM estimates with a satisfactory degree of precision the Busy Time Fraction and the traffic nature with at most 10% of errors. Although the accuracy is not perfect, we believe that FAM is a practical scheme for enabling mobile clients to crowd-source Wi-Fi performances, providing guidance to hierarchize different WLANs, and inferring the better access point selection.

For future work, we continue to address some of the open challenges in Wi-Fi network load estimation, such as exploring how the frame aggregation can infer the AP load when using other network topologies that contain a wired network.

Acknowledgments

The authors wish to thank the anonymous referees for their thorough and constructive review of an earlier version of this paper.

References

[1] Cisco annual internet report (2018–2023) white paper.

[2] K. Sood, S. Liu, S. Yu, Y. Xiang, Dynamic access point association using Software Defined Networking, in: International Telecommunication Networks and Applications Conference (ITNAC), 2015, pp. 226–231. doi:10.1109/ATNAC.2015.7366817.

[3] M. Amer, A. Busson, I. Guérin Lassous, Considering frame aggregation in association optimization for high throughput wi-fi networks, in: ACM International Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, & Ubiquitous Networks (PE-WASUN), 2018, p. 55–62. doi:10.1145/3243046.3243057.

[4] M. Li, M. Claypool, R. Kinicki, Wbest: A bandwidth estimation tool for ieee 802.11 wireless networks, in: 2008 33rd IEEE Conference on Local Computer Networks (LCN), 2008, pp. 374–381. doi:10.1109/LCN.2008.4664193.

[5] L. Song, A. Striegel, Leveraging frame aggregation for estimating wifi available bandwidth, in: IEEE International Conference on Sensing, Communication, and Networking (SECON), 2017, pp. 1–9. doi:10.1109/SAHCN.2017.7964908.

[6] A. Farshad, M. Lee, M. K. Marina, F. Garcia, On the impact of 802.11n frame aggregation on end-to-end available bandwidth estimation, in: IEEE International Conference on Sensing, Communication, and Networking (SECON), 2014, pp. 108–116. doi:10.1109/SAHCN.2014.6990333.

[7] P. Nayak, S. Pandey, E. W. Knightly, Virtual speed test: an ap tool for passive analysis of wireless lans, in: IEEE Conference on Computer Communications (INFOCOM), 2019, pp. 2305–2313. doi:10.1109/INFOCOM.2019.873759.

[8] S. Zhu, A. Mohammed, A. Striegel, A frame-aggregation-based approach for link congestion prediction in wifi video streaming, in: International Conference on Computer Communications and Networks (ICCCN), 2020, pp. 1–8. doi:10.1109/ICCCN49398.2020.9209675.

[9] B. Melander, M. Bjorkman, P. Gunningberg, A new end-to-end probing and analysis method for estimating bandwidth bottlenecks, in: IEEE Global Telecommunications Conference (Globecom), Vol. 1, 2000, pp. 415–420. doi:10.1109/GLOCOM.2000.892039.

[10] M. Jain, C. Dovrolis, Pathload: A Measurement Tool for End-to-End Available Bandwidth, in: Passive and Active Measurement Workshop (PAM), 2002, pp. 14–25.

[11] V. Ribeiro, R. Riedi, J. Navrátil, L. Cottrell, PathChirp: Efficient Available Bandwidth Estimation for Network Paths, in: Passive and Active Measurement Workshop (PAM), 2003, pp. 1–11. doi:10.2172/813038.

[12] A. Johnsson, M. Bjorkman, B. Melander, An Analysis of Active End-to-end Bandwidth Measurements in Wireless Networks, in: IEEE/IFIP Workshop on End-to-End Monitoring Techniques and Services, 2006, pp. 74–81. doi:10.1109/E2EMON.2006.1651282.

[13] J. Strauss, D. Katabi, F. Kaashoek, A measurement study of available bandwidth estimation tools, in: ACM SIGCOMM Conference on Internet Measurement (IMC), 2003, p. 39–44. doi:10.1145/948205.948211.

[14] N. Hu, P. Steenkiste, Evaluation and characterization of available bandwidth probing techniques, IEEE Journal on Selected Areas in Communications 21 (6) (2003) 879–894.

[15] V. Ribeiro, M. Coates, R. Riedi, S. Sarvotham, B. Hendricks, R. Baraniuk, Multifractal cross-traffic estimation, in: ITC Specialist Seminar on IP Traffic Measurement, Modeling, and Management, 2000, pp. 15–1.

[16] G. Aceto, F. Palumbo, V. Persico, A. Pescape, Available bandwidth vs. achievable throughput measurements in 4g mobile networks, in: International Conference on Network and Service Management (CNSM), 2018, pp. 125–133.

[17] P. Ha, E. Zhang, W. Sun, F. Cui, L. Xu, A novel timestamping mechanism for clouds and its application on available bandwidth estimation, in: IEEE International Conference on Distributed Computing Systems (ICDCS), 2019, pp. 12–22. doi:10.1109/ICDCS.2019.00011.

[18] W. E. Castellanos, J. Guerri, P. Arce, Available bandwidth estimation for adaptive video streaming in mobile ad hoc, International Journal of Wireless Information Networks 26 (2019) 218–229. doi:10.1007/s10776-019-00431-0.

[19] H. Zhao, E. Garcia-Palacios, J. Wei, Y. Xi, Accurate available bandwidth estimation in ieee 802.11-based ad hoc networks, Computer Communications 32 (6) (2009) 1050 – 1057. doi:10.1016/j.comcom.2008.12.031.

[20] P. Megyesi, A. Botta, G. Aceto, A. Pescapé, S. Molnár, Challenges and solution for measuring available bandwidth in software defined networks, Computer Communications 99 (2017) 48–61. doi:10.1016/j.comcom.2016.12.004.

[21] D. Salcedo, C. Guerrero, R. Martínez Aguilar, Available bandwidth estimation tools: Metrics, approach and performance, International Journal of Communication Networks and Information Security 10 (2018) 580–587.

[22] H. K. Lee, V. Hall, K. H. Yum, K. I. Kim, E. J. Kim, Bandwidth estimation in wireless lans for multimedia streaming services, in: IEEE International Conference on Multimedia and Expo (ICME), 2006, pp. 1181–1184. doi:10.1109/ICME.2006.262747.

[23] M. Li, M. Claypool, R. Kinicki, Packet dispersion in ieee 802.11 wireless networks, in: IEEE Conference on Local Computer Networks (LCN), 2006, pp. 721–729. doi:10.1109/LCN.2006.322028.

[24] H. Nasreddine, J. Issam, J. Maher, Ben, An accurate two dimensional Markov chain model for IEEE 802.11n DCF, Wireless Networks 24 (2018) 1019–1031. doi:10.1007/s11276-016-1383-z.

[25] B. S. Kim, H. Y. Hwang, D. K. Sung, Effect of frame aggregation on the throughput performance of ieee 802.11n, in: IEEE Wireless Communications and Networking Conference (WCNC), 2008, pp. 1740–1744. doi:10.1109/WCNC.2008.310.

[26] IEEE, Standard for information technology–telecommunications and information exchange between systems - local and metropolitan area networks–specific requirements - part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications, IEEE Std 802.11-2020 (Revision of IEEE Std 802.11-2016) (2021) 1–4379doi:10.1109/IEEESTD.2021.9363693.

[27] N. E. H. Bouzouita, A. Busson, H. Rivano, Analytical study of frame aggregation level to infer IEEE 802.11 network load, in: International Wireless Communications and Mobile Computing (IWCMC), 2020, pp. 952–957. doi:10.1109/IWCMC48107.2020.9148448.

[28] S. M. Ross, Introduction to Probability Models, tenth Edition, Academic Press, San Diego, CA, USA, 2010. doi:10.1016/C2009-0-30640-6.

[29] The network simulator ns-3, https://www.nsnam.org/.

[30] A. Massouri, L. Cardoso, B. Guillon, F. Hutu, G. Villemaud, T. Risset, J. Gorce, Cortexlab: An open fpga-based facility for testing sdr cognitive radio networks in a reproducible environment, in: IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2014, pp. 103–104. doi:10.1109/INFCOMW.2014.6849176.

[31] IEEE, Draft standard for information technology – telecommunications and information exchange between systems local and metropolitan area networks – specific requirements part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications amendment enhancements for high efficiency wlan, IEEE P802.11ax/D4.0, February 2019 (2019) 1–746.

[32] F. Abut, Through the diversity of bandwidth-related metrics, estimation techniques and tools: An overview, International Journal of Computer Network and Information Security 10 (2018) 1–16. doi:10.5815/ijcnis.2018.08.01.

[33] Ookla speedtest.
URL http://www.speedtest.net/

[34] Iperf.
URL https://iperf.fr/

A Frame transmission duration

Note that, in our work, transmission period takes into account the data packet transmission itself, as well as all protocol overhead (backoff, SIFS, DIFS and

ACK or block ACK transmission). This transmission duration is expressed as :

$$\begin{aligned} f(l) = & T_{DIFS} + T_{backoff} + T_{PHY} + T_{SIFS} \\ & + T_{Block\ ACK} + \frac{1}{frequency} * T_{Block\ ACK\ Request} \\ & + \frac{(MPDUDelimiter + MacHeader + Payload + FCS) \times 8 \times l}{Physical\ transmission\ rate} \end{aligned} \tag{9}$$

where $T_{Block\ ACK\ Request}$ means that a response is requested upon transmission of a frame whose sequence number is distant at least by a given threshold multiplied by the transmit window size from the starting sequence number of the transmit window. We compute the frequency of sending of these frames and we add its duration to the frame duration.

For the non-aggregated cross traffic, the transmission duration is expressed as:

$$\begin{aligned} g = & T_{DIFS} + T_{backoff} + T_{PHY} + T_{SIFS} + T_{ACK} \\ & + \frac{(MacHeader + Payload + FCS) \times 8}{Physical\ transmission\ rate} \end{aligned} \tag{10}$$

where $T_{backoff}$ denotes the backoff average value since the backoff time is a changing variable. Thus, we get:

$$T_{backoff} = \frac{CW_{min}}{2} \times T_{slot} \tag{11}$$

We define $T_{BlockACK}$ and T_{ACK} to be respectively the time required to send the block acknowledgment frame and the acknowledgment frame. They also count their physical header.

B AP queuing system under frame aggregation

Before the introduction of frame aggregation scheme and for a given enhanced distributed channel access (EDCA) class (an extension of the Distributed Coordination Function scheme that includes Quality of Service traffic prioritization for data packets.), the AP purses a First In First Out (FIFO) per-packet scheduling method. With frame aggregation, the aggregated transmissions act as a batch scheduler which alters the timing characteristics of received packets. In Figure 15, we illustrated the following example. It consists of an AP queue with two interleaved sequences of packets sent to two different clients, A and B, on a Wi-Fi link. Ai defines the i^{th} packet targeted to client A while Bi is the i^{th} packet destined to client B. In the case depicted in Figure 15, the packet at the head of the AP queue is addressed to A. All A's packets (within the limit of the maximum number of aggregated sub-frames) will be aggregated and sent out as an A-MPDU. Then, an aggregated frame with B's packets is sent.

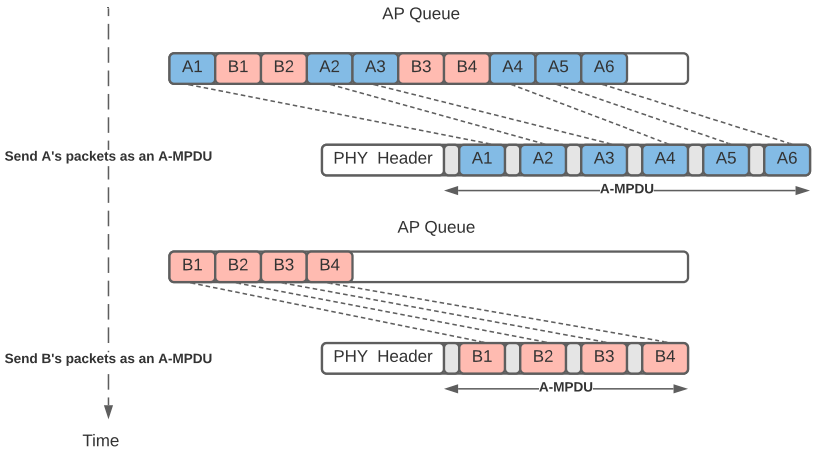


Figure 15: An Example of scheduling under frame aggregation.

C Other classes of wireless server model

Class II:Transition from state APC when presenting the transition probabilities of this class, we apply the same principle as the first class.

Contrarily to the previous class of transitions where only two possible transitions are allowed from the state APP, there are here three possible transitions under some conditions. S_{n+1} can be APP, APC, or SP.

First, we suppose that there will be another APC, the transition from APC to APC is deemed possible if and only if $X_n = i = 0$ and $Y_n = l = 0$: it is impossible to have two successive APC transmissions if $i > 0$ or $l > 0$ due to the AP queuing system detailed in B. The transition probability from APC to APC with $Y_{n+1} = m > 0$ is:

$$P_{(0,j,k,APC)(0,m,q,APC)} = P\Big(S_{n+1} = APC | S_n = APC, X_n = X_{n+1} = 0, \\ Y_{n+1} = m > 0, Z_{n+1} = q\Big) \cdot \mathbb{1}_{\lfloor \frac{T_{AC}(j)}{d_c} \rfloor = m} \cdot \mathbb{1}_{k + \lfloor \frac{T_{AC}(j)}{d_p} \rfloor = q} \tag{12}$$

Now if we suppose that the AP gains the medium access to transmit the DL probe flow, the transition probability from APC to APP is possible only if $X_n = i = X_{n+1} = l$ since the AP cannot receive probe frames in its buffer during the transmission of the cross traffic and $X_{n+1} = l > 0$. The corresponding probability is given by:

$$P_{(i,j,k,APC)(l,m,q,APP)} = P\Big(S_{n+1} = APP | S_n = APC, X_{n+1} = X_n = l > 0, \\ Z_{n+1} = q\Big) \cdot \mathbb{1}_{\lfloor \frac{T_{AC}(j)}{d_c} \rfloor = m} \cdot \mathbb{1}_{k + \lfloor \frac{T_{AC}(j)}{d_p} \rfloor = q} \tag{13}$$

If we assume that the probe traffic node gains the medium access, the transition from APC to SP is also allowed only if $X_{n+1} = l = X_n = i$ and $Z_{n+1} = q > 0$. We have:

$$P_{(i,j,k,APC)(l,m,q,SP)} = P\Big(S_{n+1} = SP | S_n = APC, Z_{n+1} = q > 0, \\ X_{n+1} = X_n = l, Y_{n+1} = m\Big) \cdot \mathbb{1}_{\lfloor \frac{T_{AC}(j)}{d_c} \rfloor = m} \cdot \mathbb{1}_{k + \lfloor \frac{T_{AC}(j)}{d_p} \rfloor = q} \tag{14}$$

Class III:Transition from state SP In order to derive the transition probabilities of the last class, we apply the same principle. Due to space limitations, we do not describe the details of the derived transition probabilities.

From the SP state, we have three possible transitions APP, APC, or SP depending on the competition for the channel resource. First, if the next transition is APP with $X_{n+1} = l > 0$, the transition probability is given by:

$$P_{(i,j,k,SP)(l,m,q,APP)} = P\Big(S_{n+1} = APP | S_n = SP, X_{n+1} = l > 0, Z_{n+1} = q, \\ Y_{n+1} = m\Big) \cdot \mathbb{1}_{l=i+k} \cdot \mathbb{1}_{j + \lfloor \frac{T_{SP}(k)}{d_c} \rfloor = m} \cdot \mathbb{1}_{\lfloor \frac{T_{SP}(k)}{d_p} \rfloor = q} \tag{15}$$

Second, if the next transition is APC with $Y_{n+1} = m > 0$, the transition probability is defined as:

$$P_{(i,j,k,SP)(l,m,q,APC)} = P\Big(S_{n+1} = APC | S_n = SP, Y_{n+1} = m > 0, Z_{n+1} = q, \\ X_{n+1} = l\Big) \cdot \mathbb{1}_{l=i+k} \cdot \mathbb{1}_{j + \lfloor \frac{T_{SP}(k)}{d_c} \rfloor = m} \cdot \mathbb{1}_{\lfloor \frac{T_{SP}(k)}{d_p} \rfloor = q} \tag{16}$$

Finally, if the next transition is SP with $Z_{n+1} = q > 0$, the transition probability is thus formulated as:

$$P_{(i,j,k,SP)(l,m,q,SP)} = P\Big(S_{n+1} = SP | S_n = SP, Z_{n+1} = q > 0, X_{n+1} = l, \\ Y_{n+1} = m\Big) \cdot \mathbb{1}_{l=i+k} \cdot \mathbb{1}_{j + \lfloor \frac{T_{SP}(k)}{d_c} \rfloor = m} \cdot \mathbb{1}_{\lfloor \frac{T_{SP}(k)}{d_p} \rfloor = q} \tag{17}$$

D Additional probabilities

$$\mathbb{P}\Big(S_{n+1} = APC | S_n = APP, Z_{n+1} = q, Y_{n+1} = m > 0\Big) = \frac{1}{2}\mathbb{1}_{q>0} + \mathbb{1}_{q=0}$$

$$\mathbb{P}\Big(S_{n+1} = SP | S_n = APP, Y_{n+1} = m, Z_{n+1} = q > 0\Big) = \frac{1}{2}\mathbb{1}_{m>0} + \mathbb{1}_{m=0}$$

$$\mathbb{P}\Big(S_{n+1} = APP | S_n = APC, X_{n+1} = X_n = l > 0, Z_{n+1} = q\Big) = \frac{1}{2}\mathbb{1}_{q>0} + \mathbb{1}_{q=0}$$

$$\mathbb{P}\Big(S_{n+1} = APC | S_n = APC, X_n = X_{n+1} = 0, Y_{n+1} = m > 0, Zn + 1 = q\Big) = \frac{1}{2}\mathbb{1}_{q>0} + \mathbb{1}_{q=0}$$

$$\mathbb{P}\Big(S_{n+1} = SP | S_n = APC, Z_{n+1} = q > 0, X_{n+1} = X_n = l, Y_{n+1} = m\Big) = \frac{1}{2}\mathbb{1}_{l+m>0} + \mathbb{1}_{l+m=0}$$

$$\mathbb{P}\Big(S_{n+1} = APP | S_n = SP, X_{n+1} = l > 0, Z_{n+1} = q, Y_{n+1} = l\Big) = \frac{1}{2} \cdot \frac{1}{2}\mathbb{1}_{q>0,m>0} + \frac{1}{2}\mathbb{1}_{q>0,m=0} + \frac{1}{2}\mathbb{1}_{q=0,m>0} + \mathbb{1}_{q=0,m=0}$$

$$\mathbb{P}\Big(S_{n+1} = APC | S_n = SP, Y_{n+1} = m > 0, Z_{n+1} = q, X_{n+1} = l\Big) = \frac{1}{2} \cdot \frac{1}{2}\mathbb{1}_{q>0,l>0} + \frac{1}{2}\mathbb{1}_{q>0,l=0} + \frac{1}{2}\mathbb{1}_{q=0,l>0} + \mathbb{1}_{q=0,l=0}$$

$$\mathbb{P}\Big(S_{n+1} = SP | S_n = SP, Z_{n+1} = q > 0, X_{n+1} = l, Y_{n+1} = m\Big) = \frac{1}{2}\mathbb{1}_{m+l>0} + \mathbb{1}_{m+l=0}$$