University of Cape Town

South Africa

A Hardware Testbed for Measuring IEEE 802.11g DCF Performance

A dissertation submitted in partial fulfillment

of the requirements for the degree

MSc in Computer Science

by

Andrew Symington

April 2009

I know the meaning of plagiarism and declare that all of the work in the document, save for that which is properly acknowledged, is my own.

This document was typeset using an adapted version of the UCLA Thesis Style for LaTeX, which may be freely downloaded from http://www.isi.edu/~johnh/SOFTWARE/uclathes.html

Table of Contents

List of Figures

List of Tables

Acknowledgments

Thank you to the following people:

- To Trish, Stuart, Gail and Dean - thank you for always being such a loving and supportive family.
- To Kerry - thank you for never letting me lose faith in my academic ability, listening to me ramble on about Markov models and for loving me even though I am now officially a geek.
- To Prof. Kritzinger - thank you for your intellectual and administrative contribution to this dissertation and for the financial support you sourced for me over the last two years.
- To Prof. Iazeolla - thank you for kindly giving up your time to discuss methods of extracting workload model parameters from wireless flow-level traces.
- Prof. Tim Dunne - thank you for taking the time to understand my experiments and for your valuable contribution to the statistical methodology used in this dissertation.
- Prof. Giuseppe Bianchi - thank you for assisting me with details of your analytic model and for putting me in contact with Ilenia, who kindly supplied me with related software.
- To Steve, Carl, Hayley, Hilton, Charles, Alapan, Hans, Paolo, Dave, Ash, Jannie, Ian, Simon and the rest of the postgraduate computer scientists at UCT - thank you for all the good times in and out of the laboratory. You all helped made Masters that much more worthwhile.
- To the Schrire and Robinson families - thank you for accommodating my dissertation experiments at your respective properties for several days at a time.
- To Eve Gill and the secretaries - thank you for all the hard work, patience and dedication that you put into administering ours and other research groups in the computer science department.
- To David Johnson - thank you kindly for introducing me to the Meraka test bed, for discussing wireless network operation at length and for sharing your unplublished research ideas with me.

Abstract of the Dissertation

A Hardware Testbed for Measuring IEEE 802.11g DCF Performance

by

Andrew Symington

MSc in Computer Science

University of Cape Town, South Africa, April 2009

The Distributed Coordination Function (DCF) is the oldest and most widely-used IEEE 802.11 contention-based channel access control protocol. DCF adds a significant amount of overhead in the form of preambles, frame headers, randomised binary exponential back-off and inter-frame spaces. Having accurate and verified performance models for DCF is thus integral to understanding the performance of IEEE 802.11 as a whole. In this document DCF performance is measured subject to two different workload models using an IEEE 802.11g test bed.

Bianchi proposed the first accurate analytic model for measuring the performance of DCF. The model calculates normalised aggregate throughput as a function of the number of stations contending for channel access. The model also makes a number of assumptions about the system, including saturation conditions (all stations have a fixed-length packet to send at all times), full-connectivity between stations, constant collision probability and perfect channel conditions. Many authors have extended Bianchi's machine model to correct certain inconsistencies with the standard, while very few have considered alternative workload models. Owing to the complexities associated with prototyping, most models are verified against simulations and not experimentally using a test bed.

In addition to a saturation model we considered a more realistic workload model representing wireless Internet traffic. Producing a stochastic model for such a workload was a challenging task, as usage patterns change significantly between users and over time. We implemented and compared two Markov

Arrival Processes (MAPs) for packet arrivals at each client - a Discrete-time Batch Markovian Arrival Process (D-BMAP) and a modified Hierarchical Markov Modulated Poisson Process (H-MMPP). Both models had parameters drawn from the same wireless trace data. It was found that, while the latter model exhibits better Long Range Dependency at the network level, the former represented traces more accurately at the client-level, which made it more appropriate for the test bed experiments.

A nine station IEEE 802.11 test bed was constructed to measure the real world performance of the DCF protocol experimentally. The stations used IEEE 802.11g cards based on the Atheros AR5212 chipset and ran a custom Linux distribution. The test bed was moved to a remote location where there was no measured risk of interference from neighbouring radio transmitters in the same band. The DCF machine model was fixed and normalised aggregate throughput was measured for one through to eight contending stations, subject to (i) saturation with fixed packet length equal to 1000 bytes, and (ii) the D-BMAP workload model for wireless Internet traffic. Control messages were forwarded on a separate wired backbone network so that they did not interfere with the experiments.

Analytic solver software was written to calculate numerical solutions for thee popular analytic models for DCF and compared the solutions to the saturation test bed experiments. Although the normalised aggregate throughput trends were the same, it was found that as the number of contending stations increases, so the measured aggregate DCF performance diverged from all three analytic model's predictions; for every station added to the network normalised aggregate throughput was measured lower than analytically predicted. We conclude that some property of the test bed was not captured by the simulation software used to verify the analytic models.

The D-BMAP experiments yielded a significantly lower normalised aggregate throughput than the saturation experiments, which is a clear result of channel underutilisation. Although this is a simple result, it highlights the importance of the traffic model on network performance. Normalised aggregate throughput appeared to scale more linearly when compared to the RTS/CTS access mechanism, but no firm conclusion could be drawn at 95% confidence. We conclude further that, although normalised aggregate throughput is appropriate for describing overall channel utilisation in the steady state, jitter, response time and error rate are more important performance metrics in the case of bursty traffic.

CHAPTER 1

Introduction

Having complete and verified performance models for wireless networks is essential to the successful deployment of the technology. The focus of this dissertation is on performance modelling of one standard for wireless networking, IEEE 802.11. This standard guides the evolution of wireless local area networks (WLANs) which are the wireless counterparts of, and possible successors to, wired Ethernet networks. More specifically, this dissertation focuses on single-hop infrastructure WLANs for Internet access, such as those found in coffee shops, homes and small businesses.

IEEE 802.11 specifies that devices coexist in the unlicensed 2.4 GHz band alongside microwave ovens and Bluetooth devices. Each band is divided up into several channels and IEEE 802.11 radio performance is heavily dependent on interference caused by neighbouring third-party transmitters. As a result of the convergence between all forms of multimedia (audio, video and data) and IT, consumers continue to place a greater demand on the underlying network infrastructure. Therefore, in addition to advances in radio technology, more of the electromagnetic spectrum is being opened to the public.

The IEEE 802.11 MAC layer adds a significant amount of overhead in exchange for regulating access to the channel and preventing collisions. Originally, this was achieved using the Distributed Coordination Function (DCF) which adds overhead in the form of delays, headers and control frames. DCF provides two major access modes, *basic access* and *RTS/CTS access*. In 2005, the IEEE 802.11e standard revision introduced the Hybrid Coordination Function (HCF). HCF is founded on the same contention-based back-off mechanism, but adds Quality of Service (QoS) support and several other protocol optimisations. Although HCF is a superior MAC protocol, its parameters are more complex and there are fewer analytic models for it. Therefore, only DCF will be considered in this work.

For fully-connected networks experiencing saturation conditions, DCF performance may be numerically predicted using one of several analytic models. Such models are typically verified through simulation for older physical (PHY) layer revisions. To the authors best knowledge, no research to date has obtained results from several analytic models and compared the results, not only amongst the models, bust also against measured results from a test bed configured in the same way. Moreover, there are only a few analyses that consider cases of non-saturation, and most make simple assumptions about the workload model to preserve analytical tractability.

1.1 Scope

This research concerns only fully-connected pure IEEE 802.11g networks that make use of the DCF channel access control mechanism. Perfect wireless channel conditions are assumed. Both saturation conditions and a synthetic workload model for Internet traffic are considered. Finally, DCF performance is measured solely by normalised aggregate throughput, which is (i) dependent on the number of stations contending for channel access, and (ii) whether the RTS/CTS access method is used.

1.2 Problem Statement

Following a survey of the literature, the following two open questions were identified as integral to understanding and measuring the performance of DCF:

1. Analytic models for saturated DCF networks are shown to provide a good fit to simulations of the same kind. However, simulations themselves make a number of assumptions about the actual system. Do these assumptions have a negligible effect and can we afford to ignore them, or is there a significant difference between analytic and measured results?

2. Although analytically convenient, saturation is an extreme load condition that seldom happens in reality. If one implements a more realistic workload model for Internet traffic, how does the performance of DCF scale compared to saturation conditions?

1.3 Contribution of This Work

This dissertation contributes the following to the area of wireless network performance modelling:

1. Two different Markov arrival processes representing packet arrival in a wireless Internet access network are implemented and compared. Also, a new cluster based model for packet length is described. For both packet arrival and length, model parameters are drawn from real traces.

2. The process of planning, constructing and configuring a prototype test bed is documented.

3. Software is written to find numerical solution for three different analytic models for DCF. Results are compared under the assumption of a common PHY.

4. Experiments are conducted on the test bed to measure saturated performance for both basic and RTS/CTS access. The measurements are then compared to the analytic results in (3) above.

5. Further experiments are conducted on the test bed to measure non-saturated performance for both basic and RTS/CTS access. The workload model described in (1) above is used and the measurements are subsequently compared to the saturation experiments.

1.4 Chapter Outline

Chapter 2 briefly covers wireless networking, the IEEE 802.11 standard and the applicable channel access control mechanisms. It continues with a description of DCF in detail, with emphasis on the various delays and timings associated with the protocol. The chapter closes with an overview of performance modelling, in which the importance of both the machine model and workload model are addressed. Chapter 3 is a survey of related literature. It is split into three broad sections, the first of which covers analytic models for measuring the performance of DCF. The second and third sections deal with workload models for Internet traffic and prototype 802.11 test beds respectively.

In Chapter 4 two different Markov models, both of which represent the arrival of packets at each client in an Internet access network, are critically compared. The Discrete-time Batch Markovian Arrival

Process (D-BMAP) is chosen as the best model for the application and is thus used in the experiments. In addition, this chapter addresses the problem of describing packet length using a k-means clustering approach that generates several log-normal packet length distributions. For both the packet length and inter-arrival models, parameters are drawn from real wireless traces.

Chapter 5 covers the design, construction and configuration of a IEEE 802.11g prototype wireless test bed, while Chapter 6 focuses on the methods used to obtain performance measurements using the test bed. In addition to describing the how the test bed is configured for the experiments, Chapter 6 also covers the experiment controller software and the steps taken to reduce external interference. The final section lists the four main experiments and their associated parameters.

Chapter 7 includes, discusses and compares the numerical results and measured results obtained from the three analytic models for saturation and four test bed experiments respectively. Conclusions are drawn in Chapter 8, followed by suggestions for future research in Chapter 9.

CHAPTER 2

Background

2.1 Wireless Local Area Networks and IEEE 802.11

The 802 family of networking standards, which includes *WiFi*, *Bluetooth*, *Zigbee* and *WiMaX*, is managed by the Institute of Electrical and Electronic Engineers (IEEE). This dissertation is concerned with IEEE 802.11[1] which is the most widely-adopted standard for Wireless Local Area Networks (WLANs).

802.11, known more popularly as *WiFi*, is a standard that defines a Physical (PHY) and Data Link (DL) layer for interoperable WLANs. The DL layer is further divided into the Media Access Control (MAC) and Logical Link Control (LLC) layers. Access to the wireless medium is coordinated by the MAC, but is dependent on a number of parameters defined in the PHY. For additional information about 802.11, consult the online version[2] of the standard itself or the interpretation by Gast [30].

2.2 IEEE 802.11 Physical (PHY) Layer

As the 802.11 standard evolved, superior modulation techniques were incorporated into the PHY (see Appendix A for a modulation compatibility list for 802.11a, 802.11b and 802.11g). Table 2.1 lists several PHY parameters that affect the operation of the MAC for five prominent modulation schemes. All values are expressed in microseconds, with the exception of the contention window.

Some transmission rates are marked as optional in the standard. Therefore, for compatibility reasons the PHY always forwards control frames at a *basic rate*, which is one of the mandatory rates supported

[1] For the sake of brevity, from this point onwards IEEE 802.11 shall be referred to simply as "802.11"
[2] Available : http://standards.ieee.org/getieee802/802.11.html (Accessed 15/07/2008)

by all stations in the network. Data is forwarded at a *full rate*, which is determined by negotiation and a rate control algorithm. The maximum MAC Protocol Data Unit (MPDU) length for all current modulation schemes is 2312 bytes. MAC Service Data Units (MSDU) can exceed this length, but are fragmented into multiple MPDUs prior to transmission. 802.11b added support for the optional *short preamble*, which reduces both the preamble and Physical Layer Convergence Protocol (PLCP) duration for High Rate DSSS (HR/DSSS) frames.

Table 2.1: List of selected 802.11 PHY implementations and their associated parameters

PHY (modulation)	Slot (σ)	SIFS	Preamble[1]	PLCP[1]	Sig. Ext.	$[CW_{min}, CW_{max}]$
IR	$8\mu s$	$10\mu s$	$16\mu s/20\mu s$	$41\mu s/25\mu s$	-	[64,1024] slots
FHSS	$50\mu s$	$28\mu s$	$96\mu s$	$32\mu s$	-	[16,1024] slots
HR/DSSS, DSSS	$20\mu s$	$10\mu s$	$144\mu s$ ($72\mu s$)	$48\mu s$ ($24\mu s$)	-	[32,1024] slots
OFDM	$9\mu s$	$16\mu s$	$20\mu s$	$4\mu s$	-	[16,1024] slots
ERP-OFDM (pure)	$9\mu s/20\mu s$	$10\mu s$	$20\mu s$	$4\mu s$	$6\mu s$	[16/32,1024] slots
ERP-OFDM (prot[2])	$20\mu s$	$10\mu s$	$20\mu s$	$20\mu s$	$6\mu s$	[32,1024] slots

Although they were removed in 802.11b, it is useful to list the parameter values for the Frequency Hopping Spread Spectrum (FHSS) PHY, as results from Bianchi's [14] widely-accepted analytic model are reliant on them. 802.11a introduced the Orthogonal Frequency Division Multiplexing (OFDM) multi-carrier modulation technique, which allowed up to 54 Mbps in the 5 GHz band.

Shortly thereafter, multi-carrier modulation was introduced to the 2.4 GHz band through the Extended Rate OFDM (ERP-OFDM) PHY. This PHY always adds a $6\mu s$ signal extension after data transmission. In *protection mode* ERP-OFDM uses one of several mechanisms to allow OFDM-modulated exchanges in the presence of 802.11b devices. In *pure mode*, ERP-OFDM devices may operate using a short slot time and contention window, as no 802.11b stations are assumed to be present.

[1]Brackets indicate values where the short preamble is used

[2]Further overhead is added by the protection mechanism (CTS-to-self, RTS-CTS-to-self or DSSS encapsulation) and depends on the basic DSSS rate and whether or not the short preamble is used.

For single carrier modulation (IR, FHSS, DSSS, HR/DSSS) the time taken to transmit the payload[3] is calculated by trivially dividing the payload $L_{bits}^{PAYLOAD}$ by the PHY rate (see Figure B.1 for frame assembly details). Multi-carrier modulation depends on the number of coded bits per OFDM symbol for the PHY rate (see Table B.1). The $L_{bits}^{SERVICE}$ and L_{bits}^{TAIL} OFDM symbol overhead values are fixed at 16 and 6 bits respectively, and the time taken $T_{\mu s}^{SYM}$ to send a symbol is $4\mu s$. The general formula for calculating microsecond transmit time $TX_{\mu s}$ to transmit data is provided by Equation 2.1.

$$TX_{\mu s} = Preamble + PLCP + \begin{cases} \frac{L_{bits}^{PAYLOAD}}{R_{Mbps}} & \text{single carrier} \\ T_{\mu s}^{SYM} \left\lceil \frac{L_{bits}^{SERVICE} + L_{bits}^{PAYLOAD} + L_{bits}^{TAIL}}{N_{bps}} \right\rceil & \text{multi-carrier} \end{cases} \tag{2.1}$$

2.3 IEEE 802.11 Media Access Control (MAC) Layer

In general, access to a wireless channel can be coordinated either at the circuit-level (e.g. FDMA, TDMA, OFDMA) or packet-level (e.g. CSMA, Token Ring). The available channel access methods depend on whether the network uses an *infrastructure* or *ad-hoc* topology. In infrastructure mode all stations communicate with one shared access point (AP). In ad-hoc mode, stations connect directly with their nearest neighbours and routing is managed by a subset of the stations in the network.

802.11 provides three channel access control protocols, all of which operate at the packet level. The mandatory Distributed Coordination Function (DCF) provides contention-based channel access, while the optional Point Coordination Function (PCF) provides contention-free channel access. Contention-free access is only supported by infrastructure networks. The Hybrid Coordination Function (HCF) was later introduced by 802.11e and provides backward-compatible enhancements to both DCF and PCF.

2.3.1 Inter Frame Spaces

The fixed delays illustrated in Table 2.2 are used by the three medium access control techniques to prioritise channel activities. All of the inter-frame spaces are derived from the PHY parameters in

[3]The payload (in bits) comprises of the data plus MAC header

Table 2.1. In the absence of inter-frame spaces, it would be possible for a third party to commence transmission in the time separating a previous data transmission and the associated acknowledgement, disrupting transmission sequences.

Table 2.2: Inter-frame spaces in IEEE 802.11 (in ascending order of duration)

Name	**Derivation**	**Description**
Short IFS (SIFS)	*Refer to Table 2.1*	Separates request from acknowledgement frames
PCF IFS (PIFS)	$\sigma + SIFS$	Separates a contention from contention-free zone
DCF IFS (DIFS)	$2\sigma + SIFS$	Separates contention-based transmission sequences
Extended IFS (EIFS)	$SIFS + ACK^4 + DIFS$	Permits any hidden nodes to acknowledge a frame
Arbitrary IFS (AIFS)	*Set by user*	Defines traffic class priority in 802.11e

2.3.2 Distributed Coordination Function (DCF)

DCF is an implementation of CSMA/CA that uses randomised exponential binary back-off and a virtual carrier to coordinate access to a common channel. Time is discretised into σ microsecond units. This value is specified by the PHY as long enough to accommodate propagation delay, switch the radio mode twice, perform a Clear Channel Assessment (CCA) and execute any required MAC processing.

When a new packet arrives for transmission the station selects a random number uniformly in the range $[0, CW_{min} - 1]$, which is called the *contention window*. The station is forced to wait, or *back-off*, for that number of idle slots. If at any time during back-off the channel is sensed as busy, the counter seizes until the channel is sensed idle for a DIFS period.

If the length of the data frame is less than the *RTS threshold* the transmission sequence follows *basic access*. Here, the sender transmits the data (DATA) frame immediately and, on successful reception, the receiver transmits an acknowledgment (ACK) frame in return. Otherwise, *RTS/CTS access* is used. RTS/CTS access reduces the cost of collisions, especially in networks with a large number of hidden

[4]Microsecond time required to send a 14 byte ACK frame at 1 Mbps

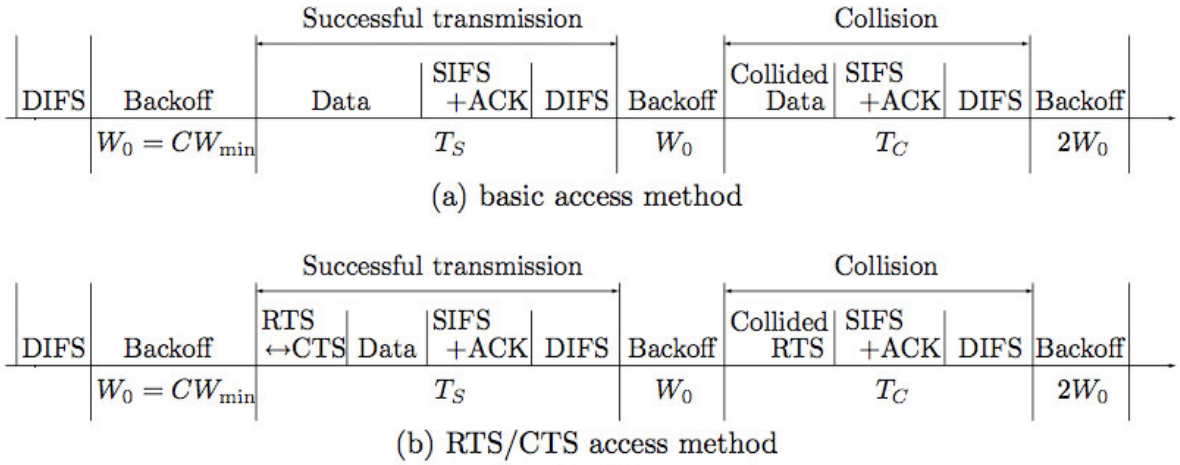

Figure 2.1: Comparison of the two access mechanisms in 802.11

or exposed nodes. Figure 2.1, taken from Park [48], shows the exact sequence of frames and delays that occur in successful and unsuccessful transmissions for both basic and RTS/CTS access. T_s and T_c denote *time taken to transmit successfully* and *time taken to transmit with collision* and are used frequently by analytic models for DCF.

Depending on the relative positions of the receivers, if two stations' back-off counters expire in the same slot their frames may collide. Owing to the fading property of wireless channels, collisions do not happen uniformly across a network and a transmitting station is unable to detect a collision with itself. Therefore, RTS, CTS and DATA frames are acknowledged by CTS, DATA and ACK frames respectively. In the case that no acknowledgement is received, the transmitting station assumes a collision. Neighbouring stations that observe the collision and defer access for an EIFS period to allow any hidden nodes to acknowledge successful receipt of the first frame. In the event of a collision, the contention window is doubled and a new random back-off is chosen uniformly in the new range. The station then backs off for the randomly chosen number of slots and then attempts to retransmit.

The binary back-off procedure continues until the contention window reaches CW_{max}. Thereafter, the contention window remains at CW_{max} until the station has retried exactly five times for basic access or seven times for RTS/CTS access. If the final retry fails the frame is dropped.

2.3.3 Point Coordination Function (PCF)

The Point Coordination Function (PCF) extends DCF to provide contention-free access to the channel. Time in the network is divided into *contention periods* and *contention-free periods*, which are coordinated by the DCF and PCF protocols respectively. The transition to a contention-free service is signaled by the Access Point (AP) in a beacon frame after the channel is sensed idle for a PIFS period. The AP is responsible for polling stations one-by-one until it issues a CF-END frame, after which the network is managed by DCF. Early in the evolution of the 802.11 standard an industry dispute resulted in PCF being marked as optional. Nowadays, nearly all vendors do not provide PCF support in their products.

2.3.4 Hybrid Coordination Function (HCF)

The Hybrid Coordination Function (HCF) introduces various Quality of Service (QoS) enhancements to the MAC. It is comprised the Enhanced Distributed Channel Access (EDCA) and HCF Controlled Channel Access (HCCA) protocols, which are extensions to DCF and PCF respectively. The primary objective of HCF is to provide support four traffic classes. In EDCA each class has its own AIFS value, which determines the starting back-off value that a station can select immediately after the channel is sensed idle; a low AIFS indicates a higher priority. Furthermore, the EIFS, CW_{min} and CW_{max} values all depend on the traffic class. Various other MAC enhancements are defined in the 802.11e revision, such as frame bursting, block acknowledgements, suppressed acknowledgements and Direct Link Setup (DLS) - the ability for stations to exchange frames directly. Like PCF, the contention-free component of HCF is marked as optional and only newer hardware supports the protocol.

2.4 Perfomance Modelling

Performance modelling involves building or deriving an abstraction of a system. Such a model is used to assess the impact of design decisions on system performance. Figure 2.2, adapted from Herzog [32], shows a high-level approach to performance modelling, which involves building a system model that produces performance metrics which may be taken as representative of the actual system.

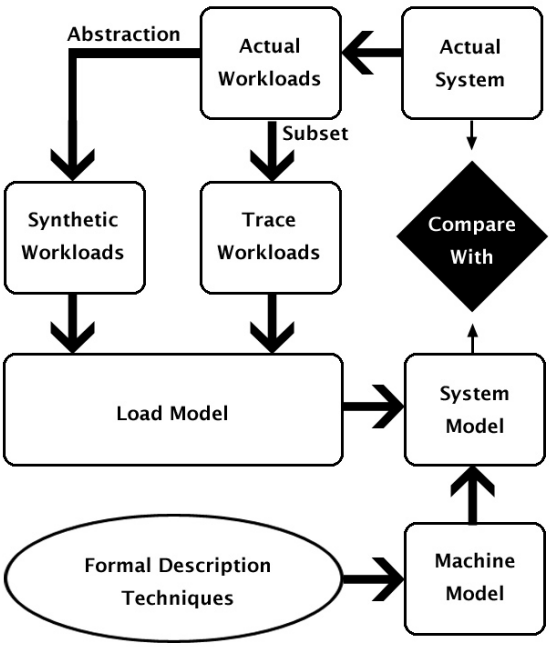

Figure 2.2: The Performance Analysis and Modelling Process

2.4.1 Performance Metrics

Performance metrics are used to quantify system performance and, in the context of networking, are closely related to Quality of Service (QoS). Common metrics for queuing networks are specified by the ITU X.605 standard and include *throughput*, *delay*, *jitter*, *queue length* and *error rate*. For wireless medium access control performance studies, it is convenient to normalise throughput (express it as a fraction of the underlying rate) to make PHYs of a different type comparable. This dissertation will only consider normalised throughput as a metric for performance.

2.4.2 Machine Models

Broadly, there are three types of machine model - *prototypes*, *simulations* and *analytic models*. A prototype is a physical copy of the system, which closely represents the operation of the system. When producing a prototype is either expensive or impractical simulation is often used. A simulator is a

digital representation of the system, which is highly configurable and scalable.

The objective of an analytic model is to obtain a closed form solution for performance metrics given a set of assumptions, which are usually more restrictive than those used in simulation or prototyping. Despite this, it is considered a superior method of modelling, as results may be calculated rapidly and for scenarios with arbitrarily extrapolated input parameters. Another advantage with analytic modelling is that it provides a deeper understanding of the system and the relation between input parameters and performance metrics. However, obtaining an analytically tractable solution for performance is often a challenging and sometimes impossible endeavour. In Chapter 3 several different analytical performance models for DCF will be discussed.

2.4.3 Workload Models

All systems are subject to a workload that broadly describes how the system is used. *Trace workloads* are replays of actual system usage and are collected over a period of time. Trace-driven models do not generalise or scale well and are therefore not appropriate for use in scenarios configured differently to the system from which the traces were originally recorded.

Synthetic workloads are abstractions from trace workloads and are usually described stochastically. The parameters for synthetic workload models are typically measured from trace data or through some analytic modelling process (for example, assuming exponential inter-arrivals). If the parameters are intuitive, they may be changed to suit the requirements of the performance model. Although difficult to derive, synthetic models provide more flexibility and demand a deeper understanding of how the system is used. They are therefore considered superior to trace replays.

CHAPTER 3

Related Work

3.1 Analytic Machine Models for DCF

In ad-hoc networks collisions do not necessarily occur uniformly across the network, complicating the modelling process. Despite this difficulty two notable performance models have been developed independently by Wang and Garcia-Luna-Aceves [63] and Holland and Vaidya [33]. However, the focus of this work will be on Internet access networks which tend to be typically managed by a central access point. To reduce the complexity of the performance model, it shall be assumed that there is full-connectivity between all stations and no direct client to client data communication takes place.

3.1.1 Bianchi's Model for DCF

Undoubtedly, the most cited performance model for DCF was published by Bianchi [14]. The model calculates normalised saturation throughput for a network of n stations. The DCF back-off procedure at each station is represented by a discrete-time Markov process with time unit equal to the 802.11 slot time σ. States are indexed by an integer set (i, j), where i represents the back-off stage and j represents the current value of the back-off counter. At each stage $(i, 0)$ the contention window W_i is chosen according to Equation 3.1 with remaining parameters taken from Table 2.1.

$$W_i = \begin{cases} 2^i CW_{min} & i < m \\ CW_{max} & \text{otherwise} \end{cases} \tag{3.1}$$

DATA and RTS frames in Bianchi's model collide consistently with constant probability p. When a previous frame is sent successfully, a new frame arrives immediately and selects a back-off value j_0

uniformly in the integer range $[0, W_0 - 1]$. The value chosen determines the state in which the back-off process begins and each value is picked randomly with uniform probability $\frac{(1-p)}{W_0}$. On every clock tick the system moves from state (i, j) to $(i, j-1)$ until $j_0 = 0$. At this point the frame transmission is attempted again. If successful, a new frame is serviced and the system restarts. Otherwise, a new back-off value j_1 is chosen using W_1 and the system moves to state $(i+1, j_1)$. This process continues up to $i = m$, where $m = log_2(\frac{CW_{max}}{CW_{min}})$. after which the contention window remains at W_m until the frame is sent successfully. A graphical depiction of the Markov chain is shown in Figure 3.1, which is adapted from Bianchi [14].

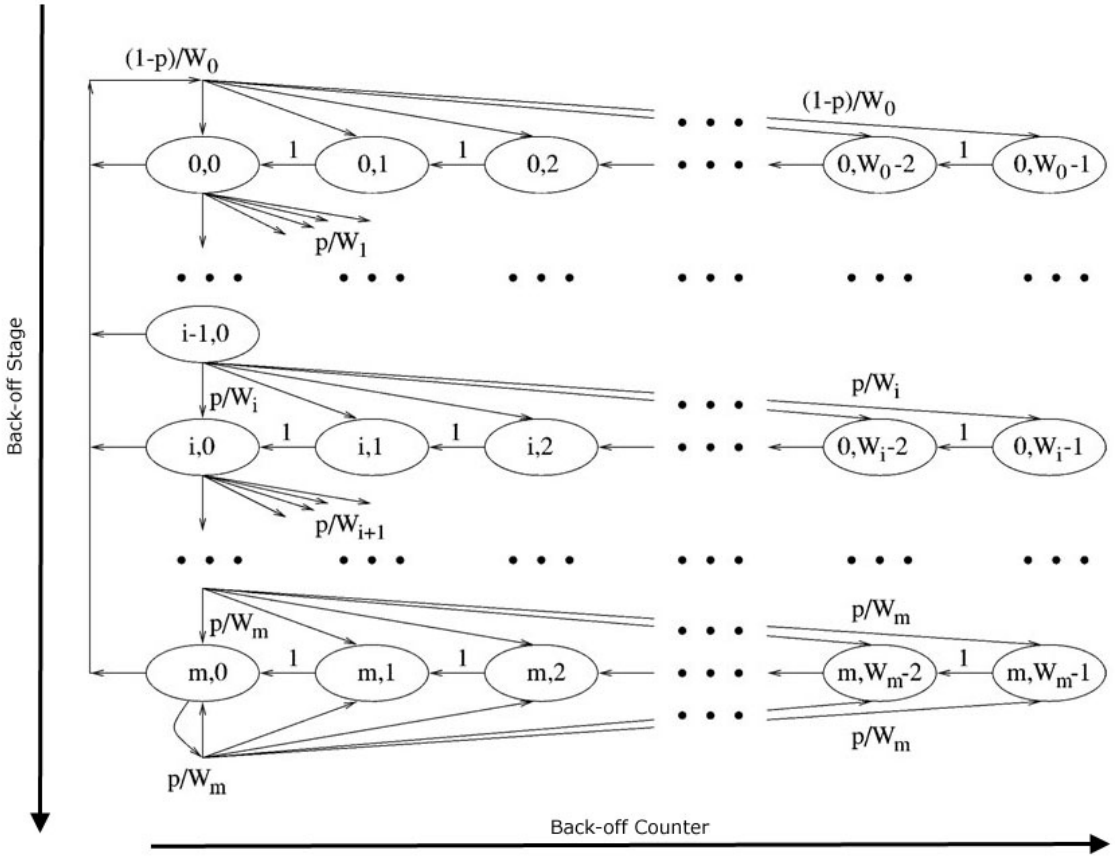

Figure 3.1: Bianchi's Markov model for DCF binary exponential back-off

A numerical solution for p and τ (the probability that a station transmits in an arbitrary slot) is

found using the two non-linear Equations 3.2 and 3.3. Equation 3.2 is derived from the previously described Markov process and Equation 3.3 is a relation stating the per-slot collision probability p is simply one minus the probability that all other stations do not transmit.

$$\tau = \frac{2}{1 + CW_{min}(1 + p\sum_{i=0}^{m-1}(2p)^i)} \tag{3.2}$$

$$p = 1 - (1-\tau)^{n-1} \tag{3.3}$$

$$P_{tr} = 1 - (1-\tau)^n \tag{3.4}$$

$$P_s = n\tau(1-\tau)^{n-1} \tag{3.5}$$

The calculated τ value is used to determine the probability P_{tr} that any station transmits and, having done so, the probability P_s that the transmission succeeds. Equations 3.4 and 3.5 express these probabilities respectively. If the expected time to transmit only the data portion of the frame is denoted as $E_{\mu s}[P]$, normalised saturation throughput S may be calculated as per Equation 3.6. Note that the values for T_c and success T_s are taken from Figure 2.1.

$$S = \frac{P_{tr}P_sE_{\mu s}[P]}{(1-P_{tr})\sigma + P_{tr}P_sT_s + P_{tr}(1-P_s)T_c} \tag{3.6}$$

In a later publication [16] Bianchi derives several extensions to the original model and provides a simpler derivation for τ. Furthermore, the work uses Little's result [42] to provide an analytic solution for delay, which is equivalent to the solution provided by Chatzimisios *et al* [19]. Expected delay $E_{\mu s}[D]$ is equal to the product of the expected number of slots $E[X]$ before a frame is transmitted and the expected length $E_{\mu s}[L]$ of a slot. The derivation for expected delay is shown in Equations 3.7 to 3.9.

$$E_{\mu s}[D] = E[X] \cdot E_{\mu s}[L] \tag{3.7}$$

$$E_{\mu s}[L] = (1-P_{tr})\sigma + P_{tr}P_{ts}T_s + P_{tr}(1-P_s)T_c \tag{3.8}$$

$$E[X] = \sum_{i=0}^{m-1}\left(p^i \cdot \frac{W_i+1}{2}\right) + \left(\frac{p^m}{1-p} \cdot \frac{W_m+1}{2}\right) \tag{3.9}$$

In summary, Bianchi's model assumes:

1. All frames collide equally with constant probability p.

2. There are a finite number of fully-connected stations.

3. Wireless channel conditions are perfect.

4. No stations experience a post-collision EIFS period. However, this can be changed by modifying a single parameter.

5. The back-off process does not seize when the channel is perceived as busy.

6. There are an infinite number of retries on the final back-off stage.

7. All stations have a packet to transmit at all time (saturation conditions).

3.1.2 Extensions to Bianchi's model

Both Ziouva and Antonakopoulos [69] as well as Vishnevsky and Lyakhov [61] present independent extensions to Bianchi's model which account for the back-off *seizing* effect of DCF. In Bianchi's model the sojourn time on each back-off stage is σ. However, in DCF the back-off counter is suspended when a neighbouring station transmits. The [69] model assumes that the number of time units spent on each slot is geometrically distributed with mean $\frac{1}{p_b}$, where p_b represents the probability that, for any slot, the channel is busy. Figure 3.2 illustrates the change to Bianchi's Markov process to accommodate back-off suspension. In [61], a far more complex method of calculation is used and results are compared with simulations of the same kind.

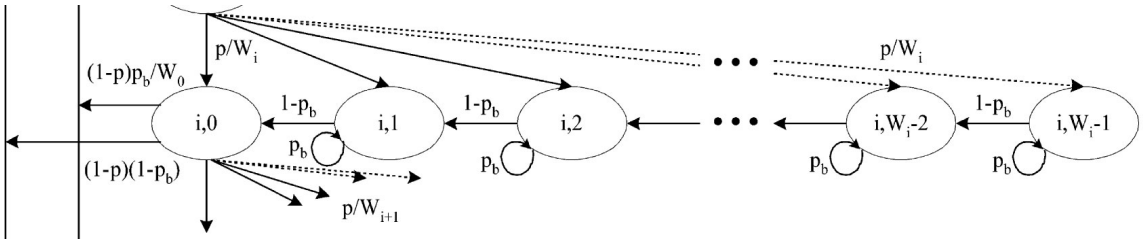

Figure 3.2: Ziouva's extension for back-off counter suspension

Another criticism of Bianchi's model is that it assumes infinite retries on the final back-off stage. Wu *et al* [65] present an extension to Bianchi's model that accurately implements the retry mechanism used in the standard. In the model the authors denote the maximum retries with contention window doubling as m' and the absolute maximum[1] number of retries as m (the change to the final stage in the Markov chain is shown in Figure 3.3). The calculation for τ is complex, but the results are more accurate when compared with simulations of the same kind.

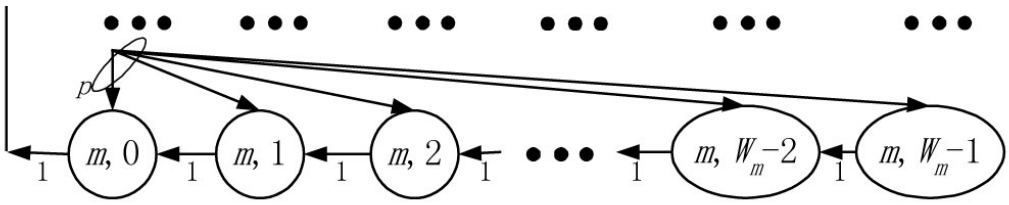

Figure 3.3: Final back-off stage in Wu's extension for finite retries

Daneshgaran *et al* [22] and Zheng Lu [68] provide independent extensions to Bianchi's model that accommodate noisy channels. They both consider only data frame corruption adhering to a noise model with a constant bit error rate (BER). Vishnevsky and Lyakhov [60] extend the model to cater for the corruption of acknowledgement frames. Ni *et al* [45] provide a corrected version of this model, arguing that the frame sequences do not adhere strictly to the 802.11 standard.

Xiao [66] proposes a performance model for the Enhanced Distributed Channel Access (EDCA) scheme used by the HCF. Robinson and Randhawa [54] propose corrections to the frame exchange sequences and timings in this model. Since 802.11e is not covered in this document, neither the derivation nor the features of either of these models are discussed.

3.1.3 Unified Machine Model for DCF in IEEE 802.11g (ERP-OFDM) Networks

Szczypiorski and Lubacz [57] present a unified analytic model for DCF in 802.11g (ERP-OFDM) networks, which is based on Bianchi's model with extensions for back-off seizure, noisy channels and finite

[1]The maximum retry count depends on whether basic or RTS/CTS access is being used

retransmissions. All extensions were taken from either Ni *et al* [45] or Wu *et al* [65]. Tabulated normalised saturation throughput results are provided as a function of the number of contending nodes for various combinations of bit-error rate, transmission rate and frame length.

3.1.4 Protocol Enhancements and Replacement Models for DCF

Several authors [21, 65, 50] propose enhancements to the actual DCF protocol. Although many of the models are shown analytically to provide superior performance, the focus of this research is on measuring the performance of DCF in real Internet access networks. Such models are mostly academic exercises and are therefore not discussed.

3.1.5 Contribution of This Dissertation

The literature provides a number of different analytic machine models for DCF. In most cases the authors verify their models by comparing results to those from simulation. What appears to be lacking is a comparison amongst the popular analytic models, specifically for the more modern OFDM PHYs. The first objective of this dissertation is to implement an analytic tool for several models [14, 65, 69, 57] and compare the resultant normalised throughput for a 54 Mbps OFDM network as a function of the number of contending stations, assuming perfect channel conditions and full saturation.

In addition to a comparison of analytic models results, a similar test bed experiment was conducted for one to eight contending stations. The normalised throughput results from the analytic models were compared directly to the test bed experiments and are presented later in this document.

3.2 Workload Models for IEEE 802.11

A saturation workload model is a theoretical system state in which all stations have a packet available for transmission immediately after the previous packet has been serviced. It is a method of stress-testing the machine model in a way that is usually convenient to solve analytically. Realistically, an operational network might experience saturation conditions only for very brief moments of time.

3.2.1 Hierarchical Model for TCP/IP Traffic

A popular approach to modelling wide area traffic is based on the principle behind TCP, which dominates all other protocols over wide-area IP networks. When a new connection is made in the network a *session* is established. Within a session *flows* arrive at a certain rate; a flow can be thought of as a collection, or burst, of *packets*. Figure 3.4, taken from Muscariello *et al* [44], shows a packet arrival process based on this paradigm.

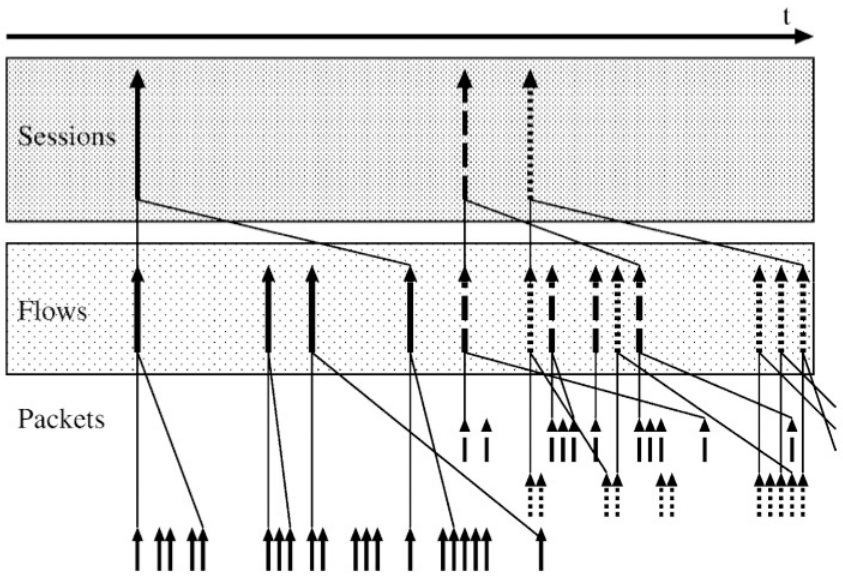

Figure 3.4: The Session/Flow/Packet paradigm

3.2.2 Workload Models for Packet Arrival

Building analytically tractable models for the Internet is a complex and open problem, as Internet usage is both immensely diverse and prone to rapid change [27]. If one plots sample Internet usage over several days it is clear that it cannot be modelled by a stationary stochastic distribution. There are patterns that emerge as a function of the underlying users *viz.* more traffic during daylight hours, less on weekends. However, one is seldom interested in the performance of a system under light load,

so a large number of patterns are irrelevant. Roberts [53] suggests that if one considers a sufficiently short window period during peak usage times, then traffic can be well-approximated by a stationary stochastic function.

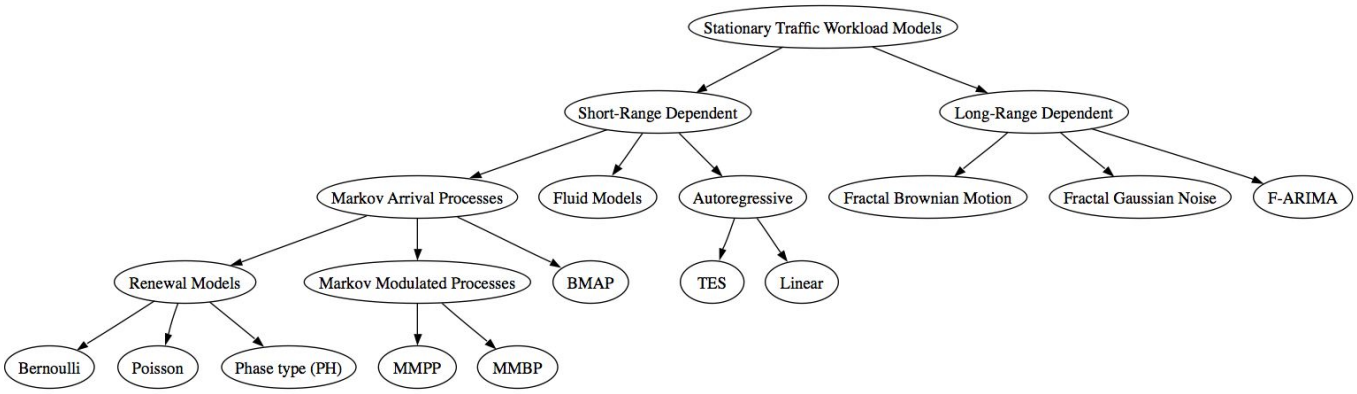

Figure 3.5: Taxonomy of Internet Traffic Workload Models

Figure 3.5 provides a taxonomy of the most popular methods used to model Internet traffic, adapted from literature surveys by Frost and Melamed [28] and Adas [10]. One may broadly divide models into two categories - those which exhibit long-range dependency (LRD) and those which do not. Internet traffic has been widely shown to exhibit LRD. LRD is an important property associated with self-similar processes, which have fractal-like patterns that repeat over different times and scales. Leland *et al* [41] show that Internet traffic displays high levels of self-similarity. Shortly thereafter Paxson and Floyd [49] showed that the widely-used memoryless Poisson model fails to capture the self-similar properties of wide area network traffic. This work invited a wide range of new models for wide area traffic.

Muscariello *et al* [44] highlight the fact that several unequal definitions for LRD exist. One popular metric for LRD is the Hurst parameter [34], which is a real number H that describes whether data has a strong tendency over time to regress to the mean or cluster in a particular direction. The H value is non-deterministic and calculated from sample generated traffic. H values close to 0.5 are considered short-range dependent, whereas those closer to 1 are considered said to be long-range dependent.

3.2.2.1 Non-Markovian Models with Short-range Dependence

Fluid traffic models are useful for instances where the difference in packet size has no significant influence on system performance [28]. This is true in Asynchronous Transmission Mode (ATM) networks, where the packet unit size is extremely small relative to the available network capacity. In contrast, WLAN packets are relatively larger (owing to there being more required overhead per packet) and the maximum capacity is significantly lower. Therefore, the size of a packet has more influence over network operation, which makes fluid modelling inappropriate for 802.11. The Linear and Transform-Expand-Sample (TES) autoregressive models are primarily used to model variable bit rate (VBR) video traffic, which is gaining popularity in commercial wide area networks. The models are based on the premise that, for most of the time, the detail change between data frames is relatively low [28].

3.2.2.2 Fractal-based Models with Long-range Dependence

Models using fractional Gaussian noise [64], fractional autoregressive integrated moving average (F-ARIMA) or fractional Brownian motion [46] exhibit long-range dependency and have been used to accurately represent wide area traffic traffic. Although such techniques provide a superior fit for Internet traffic, it is difficult to find analytically tractable solutions for them [10]. Karagiannis *et al* [38] further highlight the difficulty in interpreting metrics that describe the LRD exhibited by such models.

3.2.2.3 Markovian Arrival Processes

The Poisson arrival process is a simple and widely-used memoryless stochastic model which describes the number of discrete events (packet arrivals) that occur within a period of continuous time. The inter-arrival time between events is exponentially distributed with mean λ. The probability of k events occurring over duration τ is given by Equation 3.10.

$$Pr[N(\tau) = k] \quad = \quad \frac{e^{-\lambda\tau}(\lambda\tau)^k}{k!}, k = 0, 1..., \tag{3.10}$$

The discrete-time counterpart of the Poisson process is the Bernoulli process. In every time slot there is a probability p that the system will change to another state. The time between events is geometrically

distributed and the probability that an event occurs after n time steps is given by Equation 3.11.

$$Pr[N = n] \quad = \quad p(1-p)^{n-1} \tag{3.11}$$

Paxson [49] shows that the simple Poisson process does not provide a good fit for wide-area traffic, as it produces traffic with a very low degree of self-similarity. Another failing point of the Poisson process is that, by itself, it does not produce *bursty* traffic. Modern Internet usage is characterised by short flows (or bursts) of traffic, rather than a continuous trickle. This has become even more apparent with streaming media services, where there is a significant difference between idle and busy states.

A Markov Modulated Poisson Process (MMPP) is a system comprising of a number of states. Each state is associated with an independent mean event (normally packet) arrival rate. The movement between these states is modulated by an underlying continuous time Markov Process. MMPP models can vary substantially in complexity, depending on which traffic properties they aim to capture. The discrete-time counterpart of the MMPP is the Markov Modulated Bernoulli Process (MMBP), in which sojourn time is geometrically distributed.

Muscariello *et al* [44] propose a two dimensional MMPP model, which captures the complex influence of sessions and flows on packet arrivals. The model is fully parameterised by five variables, two of which are measured directly from traces. The remaining three parameters are estimated such that sample simulation and traces exhibit the same degree of LRD. The LRD is measured using the Hurst parameter, which is calculated using a wavelet estimator developed by Abry and Veitch [9]. The convergent method of parameter selection ensures that the resultant synthetic trace exhibit a form of pseudo-LRD, while retaining a close statistical relationship to the original traces.

The Batch Markovian Arrival Process (BMAP) proposed by Lucantoni *et al* [43] is a generalisation of the MMPP, in which batches of events occur on each state of the underlying Markov process. The parameters required to build a BMAP model cannot be directly drawn from IP traces and therefore need to be estimated. Klemm *et al* [39] make use of the expectation-maximisation algorithm to measure parameters for the BMAP model. It was found that the BMAP model provided a superior statistical fit (mean, variance, skewness, kurtosis of total bytes at various scales) to the Poisson and MMPP models, but only marginally more LRD than the MMPP traffic.

3.2.3 Workload Models for Packet Length

Although several authors have studied packet size distribution for particular traffic types, there appears to be no consensus on a general distribution that closely represents Internet packet size. Musciarello [44] observed a trimodal distribution of trace traffic and attributed it to characteristics of the TCP. The first peak is associated with network management frames, including TCP acknowledgements, which are very short by design. The other peaks occur due to applications setting the packet size equal or close to the maximum transmission unit (MTU) in order to obtain maximum network performance.

$$f(x;\mu,\sigma) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{(\ln(x)-\mu)^2}{2\sigma^2}} \tag{3.12}$$

Antoniou *et al* [11] aggregates Internet traces into windows of 1ms, 10ms, 1s and 10s and shows that, as the window size increases, so the total bytes becomes well-approximated by the log-normal distribution (shown in Equation 3.12). A comprehensive study by Walters [62] on the nature of World Wide Web (WWW) traffic indicates a further tendency towards log-normality.

3.2.4 Analytic Workload Models for IEEE 802.11

Duffy *et al* [25] and Garetto and Chiasserini [29] propose extensions to Bianchi's model in which packets arrive for transmission at the stations according to some Poisson process. Park *et al* [48] present a DCF performance model in which packet interarrival is modelled by a two-state MMPP and packet size is fixed to one of three[2] sizes. Throughput and delay are calculated using the MMPP/G/1/K queuing model. Chen and Li [20] provide an alternate extension in which packet length is governed by some arbitrary distribution, but the saturation condition is maintained. Their model is validated via simulation for 20 stations using geometrically distributed packet sizes.

3.2.5 Contribution of This Dissertation

In this dissertation two Markov packet arrival models for wireless Internet access networks were implemented and compared. Packet length was assumed to be log-normally distributed into several clusters.

[2]Recall the trimodal nature of TCP packet size in Section 3.2.3

Parameters for both the packet arrival and length models were measured from wireless Internet traces recorded at Dartmouth University in 2004. The packet length and best fitting arrival model were used to synthetically generate traffic for the test bed experiments.

3.3 WLAN Test Beds

3.3.1 Controlling Interference and Attenuation

Neighbouring 802.11 networks, microwaves, *Bluetooth devices* and cordless phones all emit electromagnetic radiation in the same 2.4 GHz Industrial Scientific and Medical (ISM) band, which causes interference and disrupts 802.11 experiments. This problem becomes especially apparent in indoor mesh test beds, where the transmit power has been purposefully attenuated to create a multi-hop network. As the SNR falls close to the Receive Signal Strength Indication (RSSI) specified by the modulation scheme, the probability of error increases. Superior strategies for multi-hop test beds involve anechoic chambers or artificially raising the noise floor, as Kaba and Raichle [37] propose, with white noise injectors. However, both strategies are significantly more expensive than fixed attenuation.

3.3.2 Manufacturer Standard Adherence

Research shows that many leading manufacturers produce and sell WLAN Network Interface Cards (NIC) that do not strictly adhere to the back-off parameters defined in the 802.11 standard. This strategy gives, as Bianchi *et al* [15] suggest, non-compliant manufacturers a competitive advantage over compliant manufacturers. The 802.11 test bed assembled as part of this research uses a reputable brand of NIC based on the Atheros AR5212 chipset.

3.3.3 Proprietary Extensions

Many manufacturers add optional proprietary extensions to their wireless products that do not form part of the 802.11 standard. Common extensions include frame bursting, compression, antenna diver-

sity, noise reduction and modification of the back-off parameters used in DCF. Such extensions are not compatible between manufacturers, so performance becomes dependent on the NIC brand implementation and not the 802.11 standard. In order to extract performance measurements that are comparable with analytic models for DCF, it is imperative that all proprietary extensions are disabled.

3.3.4 Grey Zones

In 802.11 the control rate is often set lower than the data rate. This is done to ensure that:

1. Management frames reach their intended destination with a higher probability (slower transmit rates generally make use of more robust modulation schemes)

2. The transmit rate is supported by all the stations in the network.

Stations associate at the control rate and send data to each other at another, sometimes faster, rate. This gives rise to *grey zones*; one station is in another station's grey zone if they associate, but experience significant data frame losses. To mitigate this problem rate control algorithms switch between data rates based on frame loss statistics. However, dynamic rate control is difficult to model analytically so this feature must be disabled for experimentation.

3.3.5 Academic Test Beds

Possibly the largest experimental indoor 802.11 test bed is run by the *Orbit Laboratory* at *Rutgers University* in New Jersey [51]. It is a two-tier system, where initial testing is conducted on one of ten small *sandbox* networks and final experiments are run on a 400 station network. The majority of WLAN cards used in the project are based on the Atheros AR5212 chipset, but a small subset of stations make use of Intel Pro Wireless 2915 cards.

In South Africa, the *Meraka Research Institute* in Pretoria, as part of their *Wireless Africa* research initiative, manages three outdoor and one indoor 802.11 mesh networks. The indoor network comprises of 49 stations arranged in a 7x7 grid, all of which use 802.11 Atheros NICs. Initially, Johnson [36] made

use of this test network primarily to benchmark ad-hoc routing protocols. More recently, however, the test bed is being used in a far wider range of research activities.

3.3.6 Contribution of This Dissertation

The major contribution of this research project is a critical comparison between the scalability of saturated and unsaturated DCF networks. For the unsaturated case a more realistic workload model for wireless Internet traffic is employed. Scalability was measured empirically using normalised saturation throughput readings from a nine station 802.11g test bed. The objective being to provide insight into how performance changes in the real world as a function of the number of contending stations for wireless channel access.

CHAPTER 4

The Synthetic Workload Model

4.1 Developing a Synthetic Wireless Workload Model

In order to assess the performance of real world 802.11 access networks, care must be taken to derive an Internet traffic workload model, which exhibits the following core characteristics:

1. **Intuitive** - Although regression models exist to closely exhibit the long-range dependency of Internet traffic, they tend to be less intuitive than traditional modelling techniques and do not lend themselves to analytic tractability [10].

2. **Easy to obtain parameters** - The proposed model should have parameters that may be reliably and easily measured from trace data.

3. **Accurate** - The resultant traffic generated by the proposed workload model should have a measurably good fit when compared to the trace data. Goodness of fit is assessed using four standard statistical measures at different scales, by the Hurst parameter and also visually.

4. **Repeatable** - The method used to draw model parameters should be general enough to use in different, but related, situations where wireless Internet usage is to be modelled.

Packet arrival and length were assumed to be unrelated and were therefore modelled separately. A random packet length was drawn with fixed probability from one of three log-normal distributions, each of which had parameters drawn from the packet-level traces. Packet arrival was modelled as a Markov Arrival Process (MAPs) in two different ways. This chapter concludes with a comparison of the two arrival models, indicating which provides the best fit to the trace wireless Internet traffic.

4.2 Trace Data Sets

In the early stages of model development, parameters were measured from flow-level traces provided by the *UNC/FORTH Archive of Wireless Traces, Models, and Tools* [47]. However, as noted by Muscariello *et al* [44], packet arrival has a large effect on the degree of self-similarity exhibited by Internet traffic and therefore it must be accounted for by an accurate and representative workload model. Since packet arrival information was not available in the UNC/FORTH traces, packet-level (tcpdump) traces from the *Community Resource for Archiving Wireless Data At Dartmouth (CRAWDAD)* [40] were used.

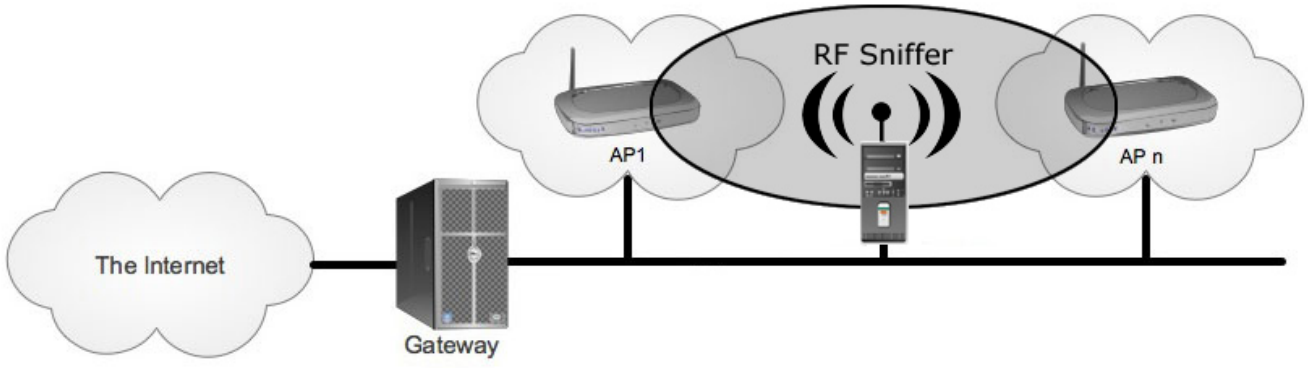

Figure 4.1: Method used to collect the Dartmouth packet-level trace data

The Dartmouth traces were collected by 18 wireless *RF sniffers* located in close proximity to various campus access points (see Figure 4.1). The MAC addresses and IPs in the tcpdump trace data is sanitized to protect the users' privacy. However, a consistent mapping between addresses and MACs is maintained, implying that client traces may be extracted individually. The chosen subset of traces was recorded by a sniffer in one of the Academic Buildings and coded *AcadBldg6-Sniffer1*. It covers the period 14:00 EST to 16:00 EST on Wednesday the 25th of February 2004.

4.3 Cluster-based Model for Packet Length

4.3.1 Objective and Model Design

Figure 4.2 shows a packet length distribution derived from the two hour Dartmouth trace. The first dominant mode is at 40 bytes and most likely represents small TCP acknowledgements and control packets. The other two dominant modes at 1300 and 1500 bytes most likely represent two MTU values used in the network. The remaining packets tend to peak around three other, significantly smaller, modes all of which are under 600 bytes. In the log-transformed packet length histogram these smaller groupings become much clearer. More interestingly, though, the three smaller grouping appear to be normally-distributed in log-space indicating that they are log-normally distributed in regular space.

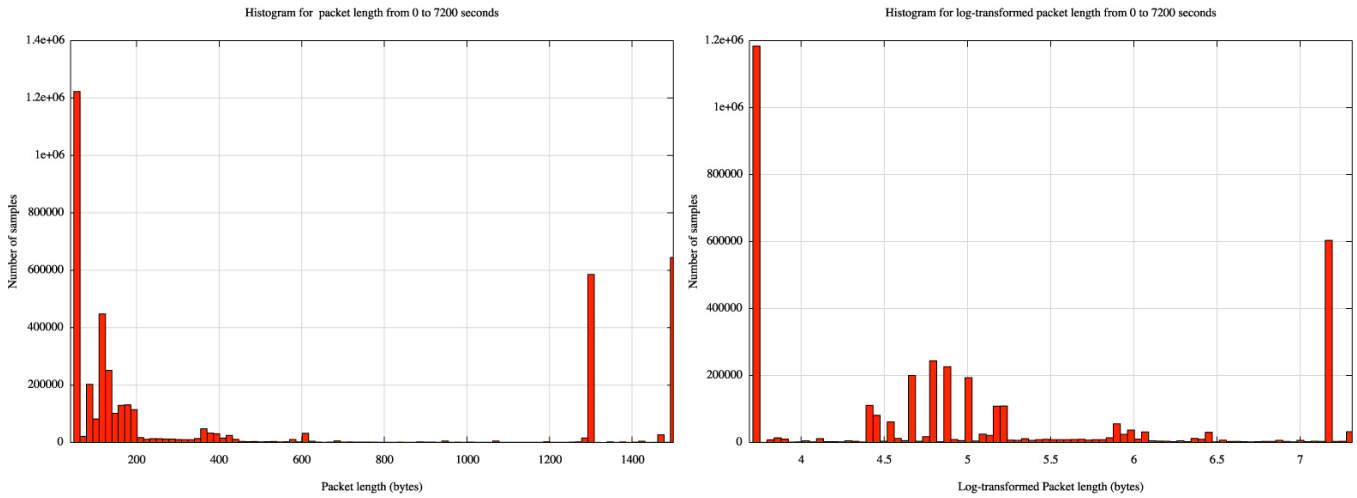

Figure 4.2: Distribution of packet length (LHS) and log-transformed packet length (RHS)

The goal of numerical clustering is to to partition the packet lengths into several similar groups. Despite clustering being an NP-complete problem [35], there are a number of heuristic algorithms available. One of the most widely-adopted algorithms is the *k-means* algorithm [31]. Compared with other methods, it is fast, simple to implement and accurate. However, depending on how well the algorithm is initialised, it suffers from convergence to local, and less optimal, solutions. Therefore, the

k-means algorithm is usually rerun several times with different seeds.

The objective of this packet length model is to best describe packet length with a minimum number of clusters. For each cluster j a log-space mean μ_j and standard deviation σ_j are measured, which represent the two parameters for an associated log-normal distribution. In addition a p_j value is also calculated for each cluster, which represents the proportion of packets that lie within the cluster. The optimal number of clusters is decided upon by analysis of the proportion of variance explained by the clustering model.

4.3.2 Obtaining Model Parameters

Two metrics are commonly used to describe error in a clustering scheme - *sum squared error* (SSE) and *sum squared total error* (SST). The former metric describes the amount of error within a cluster and is calculated in Equation 4.1 as the sum of squared differences between the cluster centroid[1] and the individual data points within the cluster. The latter metric is calculated in Equation 4.2 as the sum squared differences between the mean of the entire data set, labeled $\overline{x}$, and all the data points within a single cluster. There are N data points in total and j_i is the i^{th} data point in cluster j, which contains N_j points in total with mean $\overline{j}$.

$$SSE_j = \sum_{i=1}^{N_j} (\overline{j} - j_i)^2 \tag{4.1}$$

$$SST_j = \sum_{i=1}^{N_j} (\overline{x} - j_i)^2 \tag{4.2}$$

To assess the effectiveness of a clustering experiment the coefficient of determination (R_j^2 metric) is calculated for each cluster j as a function of SSE_j and SST_j (see Equation 4.3). This metric describes the proportion of natural variance explained by the clustering arrangement. It has an upper-bound of 1, which indicates a perfect clustering scheme. Values less than 0 indicate that the clustering scheme introduced additional variance to the data, possibly because of poor centroid choice. The final R^2 metric

[1] A centroid is the value chosen to represent all values within the cluster (usually the mean or median of all the points).

is shown in Equation 4.4 and describes the overall effectiveness of the clustering scheme. It is calculated as the average of all the R_j^2 values, weighted by the number of data points in each cluster j.

$$R_j^2 = 1 - \frac{SSE_j}{SST_j} \tag{4.3}$$

$$R^2 = \frac{1}{N} \sum_{j=1}^{k} R_j^2 N_j \tag{4.4}$$

4.3.3 Synthetic Generation of Packet Lengths

Retaining the notation from earlier in the section, let N and N_j denote the total number of data points and the number of data points in cluster j respectively. The log-space mean and standard deviation for each cluster are denoted by μ_j and σ_j. Whenever a packet is to be sent, the workload model selects a cluster j with uniform probability, weighted by $\frac{N_j}{N}$ for each cluster. The final packet length is then drawn randomly from a log-normal distribution with parameters μ_j and σ_j.

4.4 Discrete-time Batch Markovian Arrival Process (D-BMAP)

4.4.1 Objective and Model Design

The Discrete-time Batch Markovian Arrival Process (D-BMAP), depicted in Figure 4.3, is a doubly-stochastic finite state Markov arrival process, which is used to represent packets arriving for transmission at the 802.11 MAC. The time unit is chosen sufficiently short enough to accommodate, at most, one packet arrival per clock tick. In each state of the system packets arrive according to a Bernoulli process with a unique trial success probability. The first state in the chain is denoted the idle state and has a Bernoulli trial success probability of zero. An underlying discrete-time Markov chain modulates the arrival process between states over time.

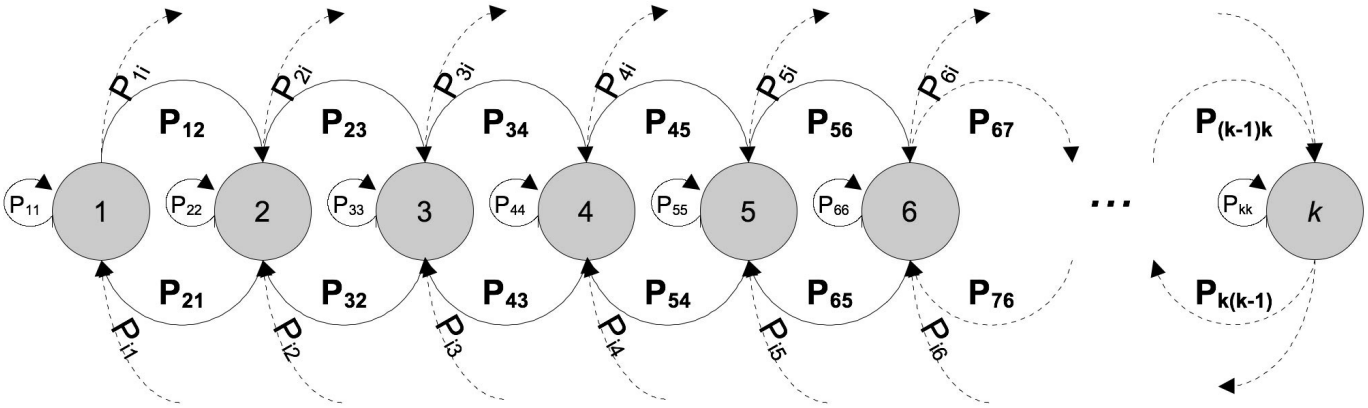

Figure 4.3: Discrete-time Batch Markovian Arrival Process (D-BMAP)

4.4.2 Parameter Measurement

The process of parameter measurement is divided into four consecutive phases - *composition*, *discretisation*, *clustering* and *parameterisation*. The trace is several hundred megabytes in size, so in order to prevent having to repeatedly open and manage the large data file a portion of the full data set is extracted in advance.

4.4.2.1 Composition

A brief analysis of several client's flows in the extracted set indicates a high degree of variance in the number of parallel flows over time (see Figure 4.4). The reason behind this is fairly straightforward - the multitasking nature of modern computers allows several Internet based applications to be active simultaneously on a single computer. For example, E-mail, web browsing and streaming radio can all be used simultaneously with packet data being transparently multiplexed at the network layer.

However, the D-BMAP model is designed such that it can only be in a single state at any point in time. Therefore, to extract parameters for such a model each client's parallel flows need to be merged to form a contiguous set of flows. This may only be done under the assumption that intra-flow packet inter-arrival follows a memoryless process. Each flow i has an associated mean rate λ_i, which is calculated as

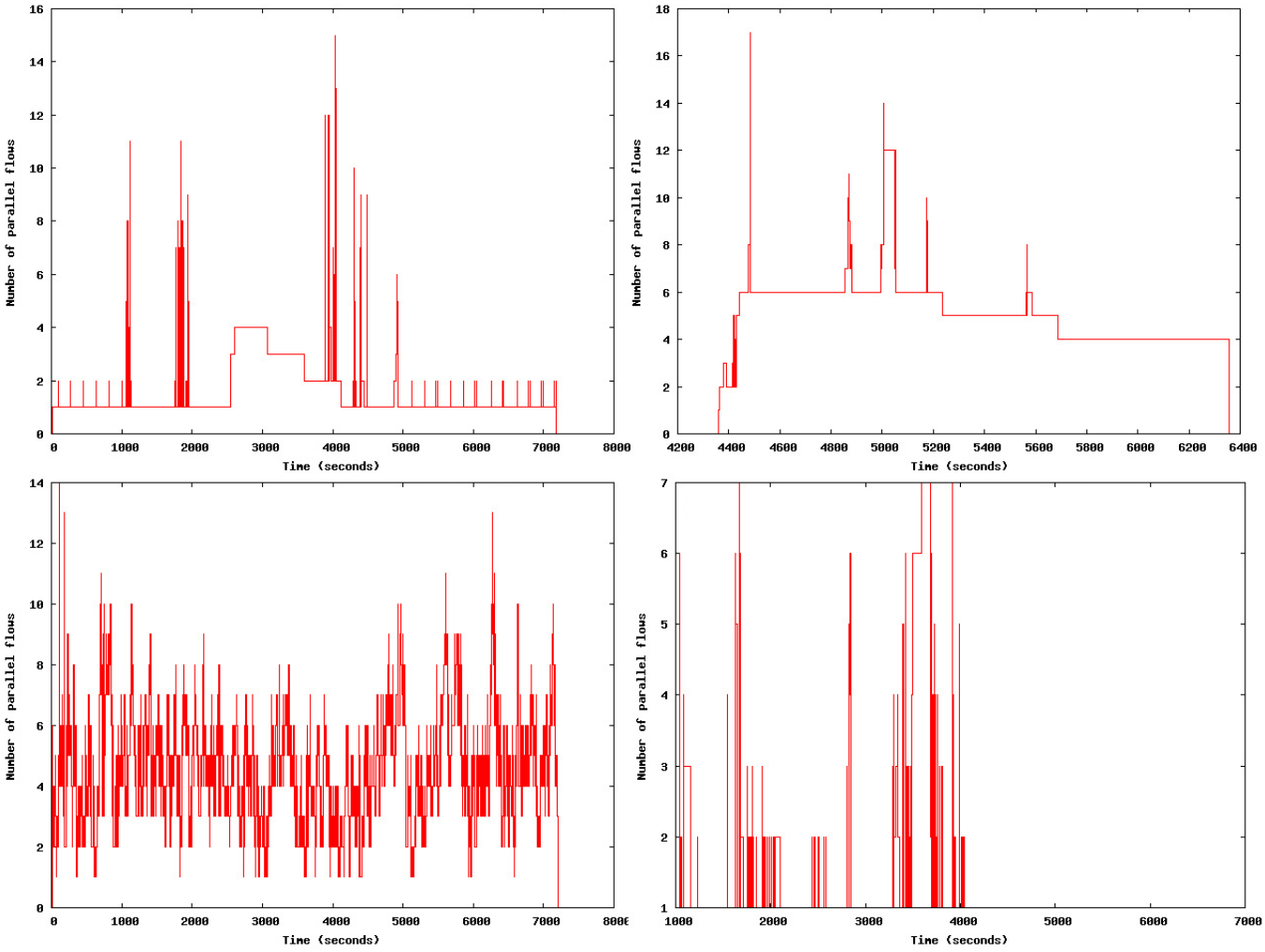

Figure 4.4: Varying degrees of flow parallelism exhibited by four randomly chosen clients

the number of packets p_i in the flow, divided by the duration d_i of the flow (see Equation 4.5).

$$\lambda_i \quad = \quad \frac{p_i}{d_i} \tag{4.5}$$

Consider Figure 4.5, which provides an example of a trivial case where a single client has two parallel flows. Flow 1 and flow 2 have associated mean rates λ_1 and λ_2 respectively. During the overlapping period $[t_1, t_2)$ the two flows merge with an arrival rate equal to $\lambda_1 + \lambda_2$. There is no overlap during $[t_0, t_1)$ and $[t_2, t_3)$ so the arrival rate remains the same. If one denotes the idle state as having $\lambda_0 = 0$ the resultant *merged flow* for this example client is comprised of five contiguous flow states of which

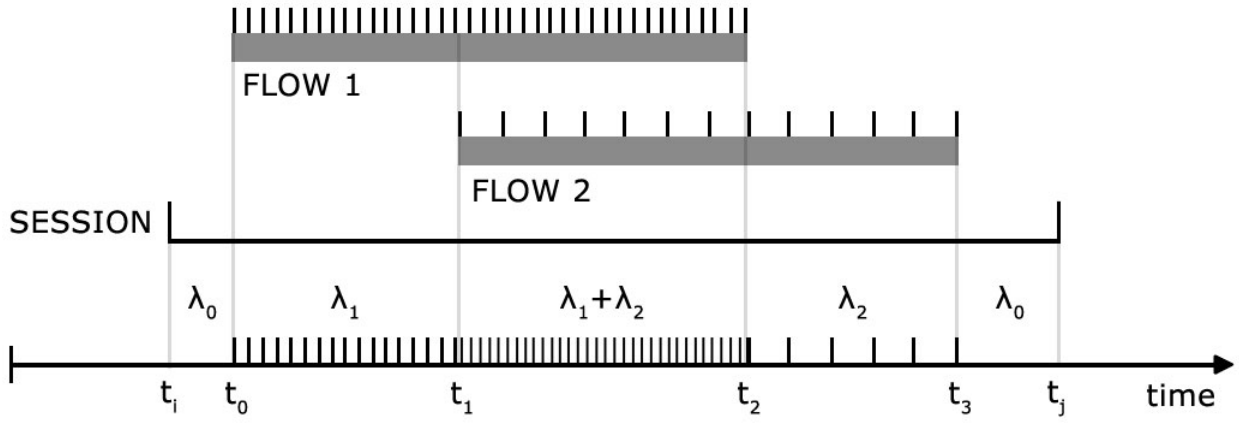

Figure 4.5: Parallel flow merging under the assumption of memoryless inter-arrival times

four are unique.

4.4.2.2 Discretisation

For 802.11 it is analytically convenient to select the D-BMAP discrete time unit equal to the PHY slot time σ. In order to discretise the data in terms of this unit, each flow's start time is first converted from its absolute representation (Unix timestamp) to the number of elapsed seconds from the start time (W_{start}) of the two hour window period, as per Equation 4.6. The significantly smaller timestamp value t'_i for each flow is then converted in Equation 4.7 to an equivalent discrete-time form T_i, which is interpreted as *the number of elapsed discrete-time ticks since the start of the window period.*

$$t'_i = \frac{t_i - W_{start}}{\sigma} \tag{4.6}$$

$$T_i = \left\lfloor \frac{t'_i}{\sigma} \right\rfloor \tag{4.7}$$

The second role of the discretisation phase is, for each merged flow i, to calculate the probability ω_i that a new packet arrives for transmission within a single clock tick. This value is interpreted as the success probability of a Bernoulli trial conducted on every clock tick. Using the equivalence between

the Poisson and Bernoulli process, the probability that there are exactly $k = 1$ arrivals in a slot σ may be calculated using the continuous-time arrival rate λ_i and Equation 4.8.

$$\omega_i = \lambda_i \sigma e^{\lambda_i \sigma} \tag{4.8}$$

4.4.2.3 Clustering

In order to measure parameters for the D-BMAP model the composed flows were first clustered into similar groups; the groups were interpreted as the states of the model. Clustering is performed as per Section 4.3 with similarity measured as the packet arrival probability per slot, ω_i. However, rather than using the R^2 metric, goodness of fit is measured using the number of packets generated by the model. For each of the experiments a sequence of packets was synthetically generated for the same duration and number of clients as in the traces.

4.4.2.4 Parameterisation

Two parameter sets are used to fully describe the D-BMAP model. The first of these is a set of k values describing the Bernoulli trial probabilities for each of the k states in the system. The first value in the set represents the idle state and thus has a value of zero. The remaining $k - 1$ values are taken as the centroids from the clustering performed in Section 4.4.2.3.

The second parameter set is a $k \times k$ transition probability matrix describing the movement between states. To generate the transition probability matrix we iterated over all client's flows and totaled the overall sojourn time for all k states in vector b. We also recorded the number of state transitions in matrix A, of size $k \times k$. Note that the diagonal of A represents the total number of visits to each state (or, alternatively, the number of flows of that state type). Using this matrix and vector as input we used Algorithm 1 to generate and normalize final transition probability matrix T.

Data: A, b, k

Result: T

begin

  for $i \leftarrow 0$ **to** $k-1$ **do**

    $A[i][i] \longleftarrow \frac{A[i][i]}{b[i]}$;

    $W \longleftarrow 0$;

    for $j \leftarrow 0$ **to** $k-1$ **do**

      $W \longleftarrow W + A[i][j]$;

    end

    for $j \leftarrow 0$ **to** $k-1$ **do**

      $T[i][j] \longleftarrow \frac{A[i][j]}{W}$;

    end

  end

end

Algorithm 1: Calculating the transition probability matrix T from measured A and b

4.4.3 Synthetic Packet Generation

Since all states in the finite and discrete-time modulating Markov chain are aperiodic the chain itself is ergodic, which means that a steady-state solution can be found. The steady-state vector Π for the underlying modulating Markov chain is calculated in advance using the normalisation equation method by Sinclair [55]. When a synthetic sequence is requested the D-BMAP generator selects a random starting state with uniform probability, weighted by the elements of Π to avoid a convergence delay.

4.4.4 Related Application Software

In the **dbmap-tools** subdirectory of **dcf-perftools** package is the source code for D-BMAP parameter estimation and simulation. First, the tcpdump [8] and Tstat [24] applications must be installed on the host computer. In addition, a suitable wireless tcpdump packet trace is required, which is arbitrarily named *tcpdump.trace* below. The process of parameter estimation begins by using the **tcpdump** and

tstat commandline utilities to extract flow and packet information from the trace:

```
$> tcpdump -r tcpdump.trace -n -tt -q tcp > packets.complete
$> tstat -Nnet.conf tcpdump.trace
```

More information about the *net.conf* file used by Tstat may be found on its official website. Copy the *log_tcp_nocomplete* and *log_tcp_complete* files from the *tcpdump.trace.out* output directory to the current working directory as *flows.incomplete* and *flows.complete* respectively. Then, create a new D-BMAP estimator configuration file called *dbmap-est.cfg* from the sample file distributed with the application software, adding the paths to the *flow.complete*, *flows.incomplete* and *packets.complete* files.

After compiling and installing the supporting applications, one estimates parameters for a D-BMAP model using the **dbmap-estimator** application in the following way:

```
$> dbmap-estimator -i dbmap-est.cfg -o dbmap-params.cfg
```

This application writes the calculated D-BMAP parameters to a file called *dbmap-params.cfg*. One can use the resultant configuration file in a prototype experiment using the **meshnet-tools** suite of applications, discussed later. Alternatively, a simple synthetic D-BMAP packet trace can be written to a file *dbmap-sim.out* using the **dbmap-simulator** application in the following way:

```
$> dbmap-simulator -o dbmap-sim.out -c dbmap-params.cfg -t 7200 -n 9
```

4.5 Hierarchical Markov Modulated Poisson Process (H-MMPP)

4.5.1 Objective and Model Design

Muscariello *et al* [44] propose a Hierarchical Markov Modulated Poisson Process (H-MMPP) model for backbone Ethernet traffic. The authors motivate that the method of parameter estimation ensures a good statistical fit and comparable degree of self-similarity when compared to the recorded traces. The model takes the following five parameters:

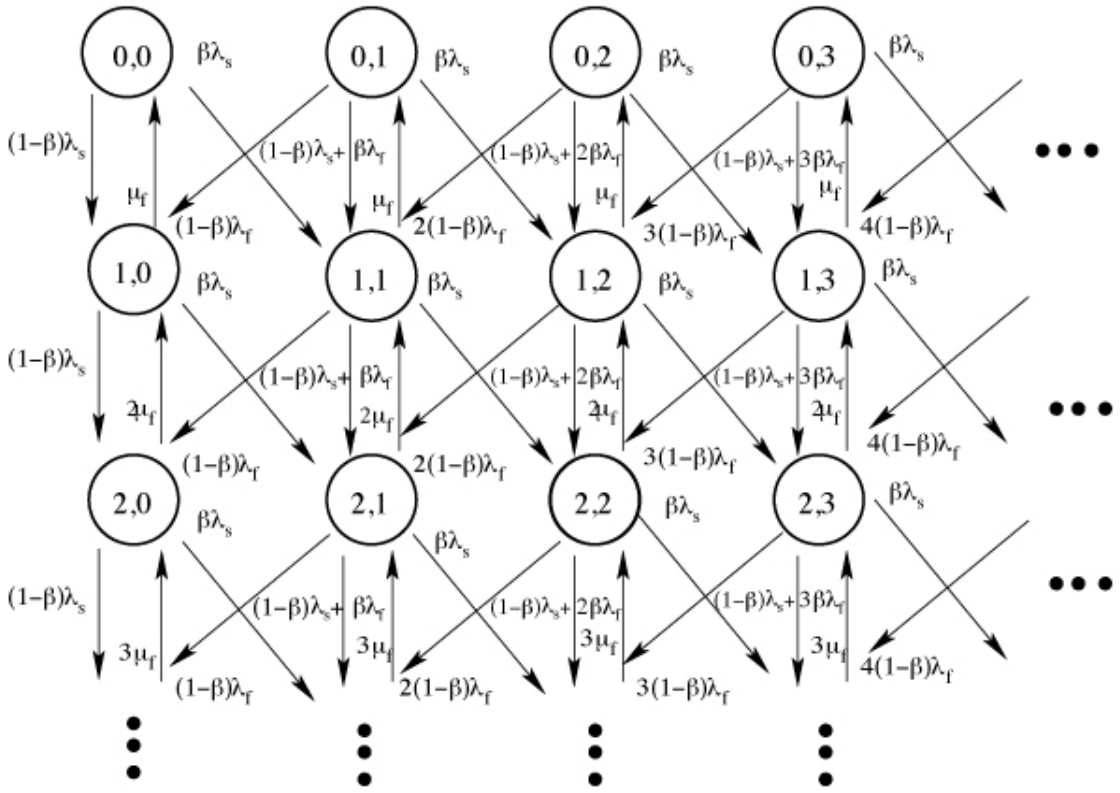

Figure 4.6: The Hierarchical MMPP Modulating CTMC, from Muscariello *et al* [44]

- λ_s : The arrival rate of sessions
- λ_f : The arrival rate of flows within a session
- λ_f : The arrival rate of packets within a flow
- N_f : The average number of total flows per session
- N_p : The average number of total packets per flow

Using the parameters above the authors construct an infinite two-dimensional continuous-time Markov chain, depicted in Figure 4.6. A single downward or upward jump indicates a new flow arrival or an existing flow termination respectively. A single right or left jump indicates the arrival of a

new session or termination of an existing session respectively. For simplicity, transition rates are written in terms of β and μ_f, which represent *the probability that a given flow is not the last one of a session* and *the rate at which a given flow terminates* respectively (see Equations 4.9 and 4.10).

$$\beta = 1 - \frac{1}{N_f} \tag{4.9}$$

$$\mu_f = \frac{\lambda_p}{N_p - 1} \tag{4.10}$$

4.5.2 Parameterisation

Values for the two parameters λ_p and N_p are measured directly from the traces. Values for three unknown parameters λ_s, λ_f and N_f are estimated indirectly using the convergence method in Algorithm 2. The algorithm relies on two additional subroutines, *MMPP* and *Hurst*. The first is an MMPP simulator, available from Mucariello's website [23], which produces a synthetic trace for a given set of parameters. The second subroutine is a wavelet-based application known as the AV estimator [9], available from Veitch [59], which calculates the Hurst parameter for a given a sequence of inter-arrival times (IAT).

First, the algorithm sets initial values for the three missing parameters. Then, it generates a synthetic trace using these parameters for the same period as the real traces. It compares the Hurst parameter for both the packet and flow IATs to those drawn from the original traces and adjusts the three missing parameters accordingly. The algorithm repeats the process until such time as the difference between the Hurst value for the actual and synthetic traces differ by less than a threshold ϵ_f and ϵ_p for both flow-level and packet level IATs respectively.

Data: $Trace, \lambda_p, N_p, \epsilon_f, \epsilon_p$

Result: $\lambda_f, \lambda_s, N_f$

begin

$\overline{H_f} \longleftarrow Hurst(Trace[FlowIAT_{ms}])$; $\overline{H_p} \longleftarrow Hurst(Trace[PacketIAT_{ms}])$;

$C \longleftarrow 1$; $N_f \longleftarrow 1$; $\frac{1}{\Lambda_f} \longleftarrow Average(Trace[FlowIAT_s])$; $fit \longleftarrow false$;

while $!fit$ **do**

$\lambda_s \longleftarrow \frac{\Lambda_f}{N_f}$; $\lambda_f \longleftarrow \frac{\lambda_s}{C}$;

$SynthTrace \longleftarrow MMPP(Trace[Time_s], \lambda_s, \lambda_f, \lambda_p, N_f, N_p)$;

$H_f \longleftarrow Hurst(SynthTrace[FlowIAT_{ms}])$; $H_p \longleftarrow Hurst(SynthTrace[PacketIAT_{ms}])$;

$fit \longleftarrow (|\overline{H_f} - H_f| < \epsilon_f) and (|\overline{H_p} - H_p| < \epsilon_p)$;

if $H_f < \overline{H_f}$ **then** $N_f = N_f + 5$;

else if $H_f > \overline{H_f}$ **then** $N_f = N_f - 1$;

if $H_p > \overline{H_p}$ **then** $C = 3 \times C$;

else if $H_p < \overline{H_p}$ **then** $C = \frac{C}{2}$;

end

end

Algorithm 2: H-MMPP convergence method for calculating λ_s, λ_f and N_f

The H-MMPP model modulates packet arrivals at the backbone level of the network. Therefore, the measured parameters are representative of aggregate traffic for 3571 clients. To be usable in the test bed experiments, the workload model must first be reduced to represent a single arbitrary client's traffic. Therefore, the final step of the parameter estimation process involves a change in representation from backbone to client traffic. The hierarchical nature of the traffic model defines packets arrivals as a function of flows and flow arrivals as a function of sessions. If one assumes that all new clients bring to the system are additional sessions, then the entire packet arrival process may be manipulated by changing the session arrival rate λ_s.

However, the factor by which λ_s must be divided is not straightforward. If one naïvely divides λ_s by the number of clients N_c in the trace, it results in a per-client session arrival rate that grossly underestimates the aggregate traffic in the network. To determine the factor K which divides λ_s to

correctly reproduce traffic, a numerical convergence method is used. K is initially set equal to N_c. Then, thirty H-MMPP simulations are run using $\frac{\lambda_s}{K}$ as the session arrival rate. The number of synthetic packets generated by each run is recorded and the average of these values N_p is calculated. If we assume that, on average, all clients generate the same number of packets over a fixed duration, the aggregate number of packets N_a is given by $N_c \times N_p$. Based on the difference between N_a and the total number of packets N_o observed in the traces, the value of K is changed and the process repeated. Convergence occurs when $|\frac{N_a - N_o}{N_o}| < \epsilon_c$, where ϵ_c determines the accuracy of the fitting procedure.

4.5.3 Synthetic Packet Generation

Like the D-BMAP model, the H-MMPP model also suffers from the steady-state convergence problem if initialised in an arbitrarily chosen state. In order to calculate a steady state vector π for the model using the same linear algebra technique as for the D-BMAP model, one has to first truncate the infinite CTMC. However, in the paper the authors present a superior analytic approach for calculating the steady state vector. They calculate that the total number of sessions and the total number of flows are both Poisson distributed with parameters δ_s and δ_f respectively (see Equation 4.11 and 4.12). Therefore, to mitigate convergence problems with the synthetic model both n_f and n_s in the starting state (n_f, n_s) are random integers drawn independently from their respective Poisson distributions.

$$\delta_s = \frac{\lambda_s \beta}{\lambda_f (1 - \beta)} \tag{4.11}$$

$$\delta_f = \frac{\lambda_s}{\beta \mu_f} \tag{4.12}$$

4.5.4 Related Application Software

In the **mmpp-tools** subdirectory of **dcf-perftools** package is the source code for the H-MMPP model parameter estimation and simulation. The method of estimating parameters is very similar to that of the DBMAP with the exception that the applications begin with a mmpp- prefix and that the configuration file for the **mmpp-estimator** application excludes a batch number and includes three threshold values for convergence - ϵ_f, ϵ_p and ϵ_c.

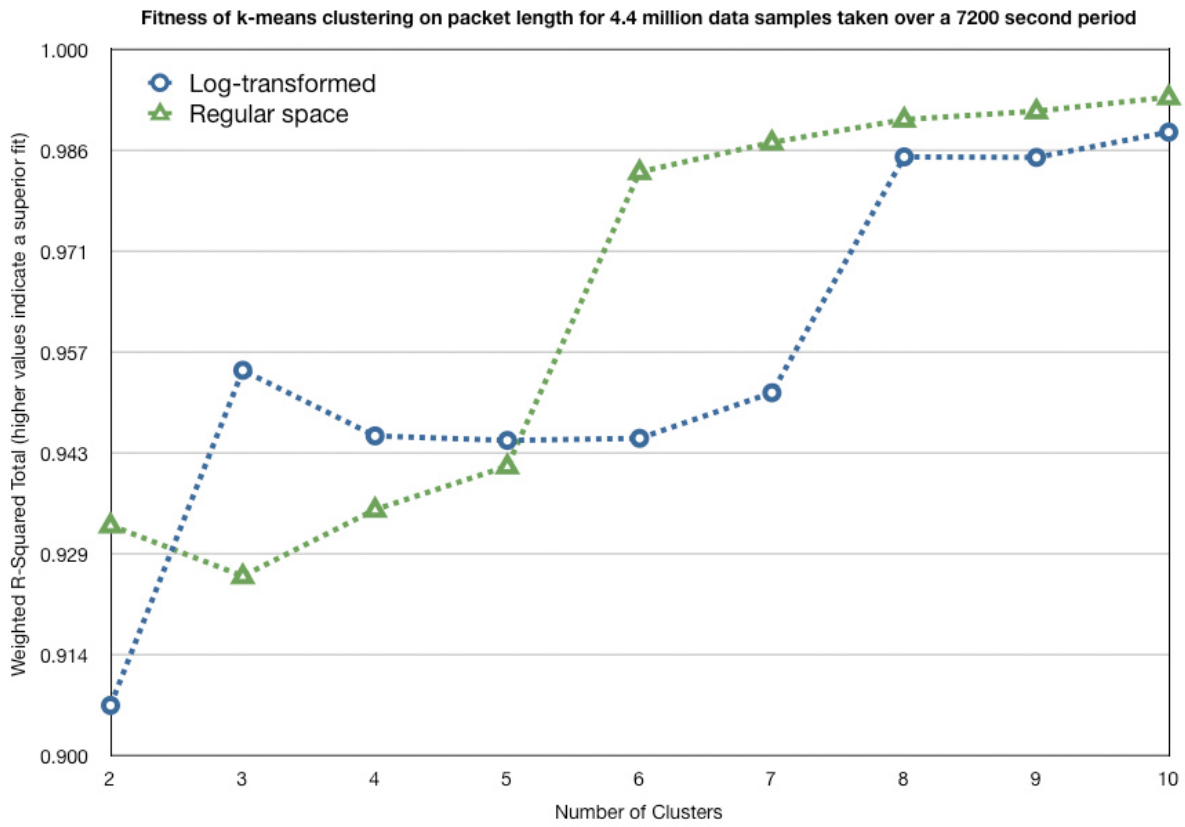

Figure 4.7: Relative Performance of the Eighteen Clustering Experiments

4.6 Assessing the Workload Model

4.6.1 Packet Length

A total of 18 k-means clustering experiments were conducted on the Dartmouth data set, which contains just over 4.4 million packet length values recorded over a two hour period. The experiments used the k-means clustering subroutine provided by Eisen Lab's *Clustering Source Code* [26]. All of the experiments were conducted on a 1.8GHz Core Duo Apple MacBook Pro. Depending on the initial partitioning, a single run of the k-means clustering algorithm might converge to local a less optimal solution. Therefore, each clustering experiment is repeated ten times and the solution that is shared by the most runs is taken as optimal. The R^2 goodness of fit metric was calculated for each clustering scheme and plotted

as a function of the number of clusters. The resulting graph is shown by Figure 4.7.

In general, a log-transform reduces the effect of outlier points on the cluster mean. With the exception of the first experiment, a log-transform benefits the clustering algorithm where there are fewer than five clusters. It is likely that this is related to the fact that there are five distinct modes in the trace packet length distribution. However, after five clusters the log-transform appears to reduce the effectiveness of the clustering by a successively smaller amount as the number of clusters increases. Although three clusters on a log-transformed data set provides a good R^2 value relative to its regular-space counterpart, resultant simulations revealed too much variance in the cluster centred on the network MTU. This results in many packets being generated with lengths well over 1600 bytes, which does not agree with the trace data. Experiment 15 produced six regular-space clusters and was chosen as the best-fit clustering for packet length, having explained over 98% of the variance in the data set.

Table 4.1: Experiment 15 : Final Six Packet Length Cluster Measurements

Metric	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6
R-Square	0.998661	0.872641	0.649549	0.997841	0.999891	0.993003
Log-space Mean	3.828926	5.889492	6.491821	7.164754	7.311727	4.903473
Log-space Std. Dev.	0.282045	0.150327	0.166223	0.03116	0.007051	0.210758
Mean	48	365	669	1294	1498	138
Data points	1511146	246181	92505	628342	678089	1245383

Table 4.1 shows summary data for each of the six clusters generated by Experiment 15. It is important to note the good fit of Clusters 1, 4, 5 and 6, indicated by a coefficient of determination value close to 1. A poor cluster fit results in greater variance in the synthetically generated data, yielding packet sizes well above and below the observed values in the trace. A further observation is that the two clusters with the worst fit, Cluster 2 and Cluster 3, describe less than 8% of the sample data.

Figure 4.8 visually compares the packet length distribution from the original traces to a synthetic trace containing the same number of samples, but generated using the cluster model and parameters

from Experiment 15. Note the consistency between the trace and the synthetic modes, as well as the proportion of packets distributed around each mode.

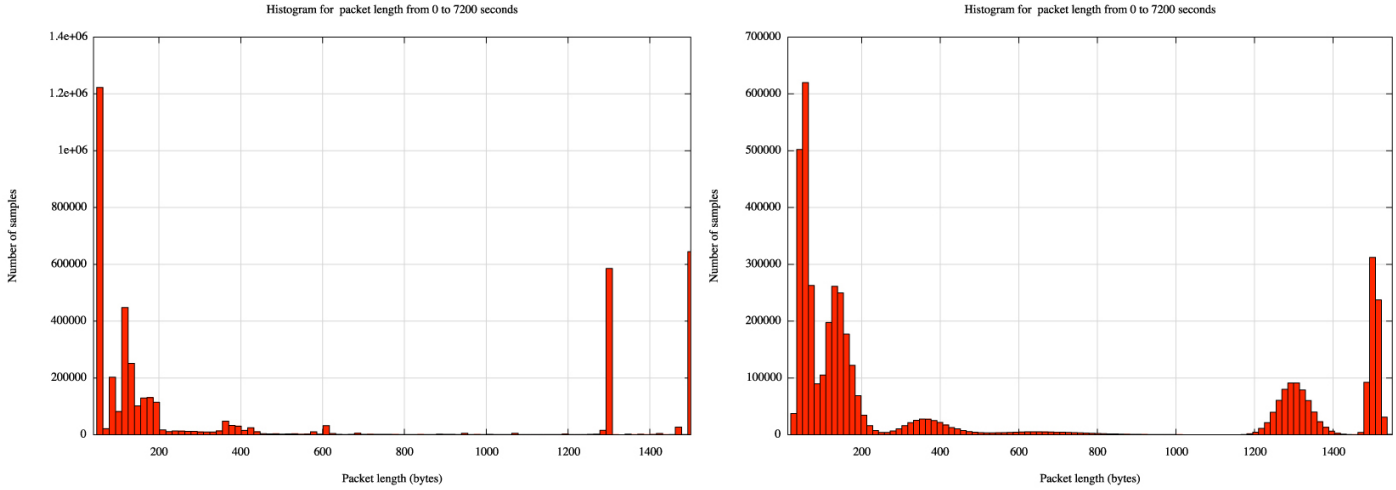

Figure 4.8: Packet length distributions for trace data (LHS) and the synthetic generator (RHS)

4.6.2 Packet Arrival

In this section results from both the D-BMAP and H-MMPP models are compared. Table 4.2 lists the measured aggregate and client-level parameters for the H-adapted MMPP model.

The number of states in the D-BMAP was chosen experimentally. Starting at five, the number of states was incremented by one until the D-BMAP produced the same number of packets measured in the traces with a 5% error. Figure 4.9 shows the relationship between the number of clusters and the aggregate number of packets generated for 3571 clients. The parameter measurement algorithm converged below the 5% threshold at 17 states and Table 4.3 lists the measured per-slot packet arrival probability for each of these 17 states. Due to the size of the data the measured transition probability matrix which modulates the 17 states is contained in the accompanying software package as **dbmap-sim.cfg**.

To assess how well the synthetic traces fitted the measured traces an analysis of aggregate traffic was

Table 4.2: H-MMPP Parameters

Parameter	Value	Interpretation
λ_s	0.585480	Backbone session arrival rate
λ_f	0.823829	Flow arrival rate per session
λ_p	3.842819	Packet arrival rate per flow
N_f	11.00000	Average number of flows per session
N_p	94.92648	Average number of packets per flow
K	633.63	Factor by which to divide λ_s to get client session arrival rate

performed at four different *scales*. In order to prevent packet length from influencing the assessment, only packet arrivals were counted. Time is discretised into into either 1, 0.1, 0.01 or 0.001 seconds, which defines the scale of the plot. The number of packets arriving in each discrete time unit was counted and plotted as a function of time.

Figures G.1 to G.2 in Appendix G compare the D-BMAP, H-MMPP and Dartmouth traces at various scales and for a various number of clients. Figure G.1 shows that the H-MMPP traffic contains short periods of intense traffic, known as bursts. This characteristic is common to Internet traffic and is not reflected in the D-BMAP plot, where the arrivals appear to follow a more memoryless process.

The method used to convert the H-MMPP aggregate parameters to client-level parameters appears to have affected the steady-state calculation. This is indicated by the overall rise in the H-MMPP plots in Figure G.4. For a small number of clients Figure G.3 suggests that the H-MMPP produces aggregate traffic that is too bursty, whereas the D-BMAP exhibits levels of burstiness closer to what is found in the traces. As more D-BMAP client traffic is merged, the aggregate plot tends towards a memoryless process. This agrees with the findings of Cao *et al* [17] with the exception that, when compared to the recorded traces, the D-BMAP model tends to memoryless at a lighter load (fewer clients).

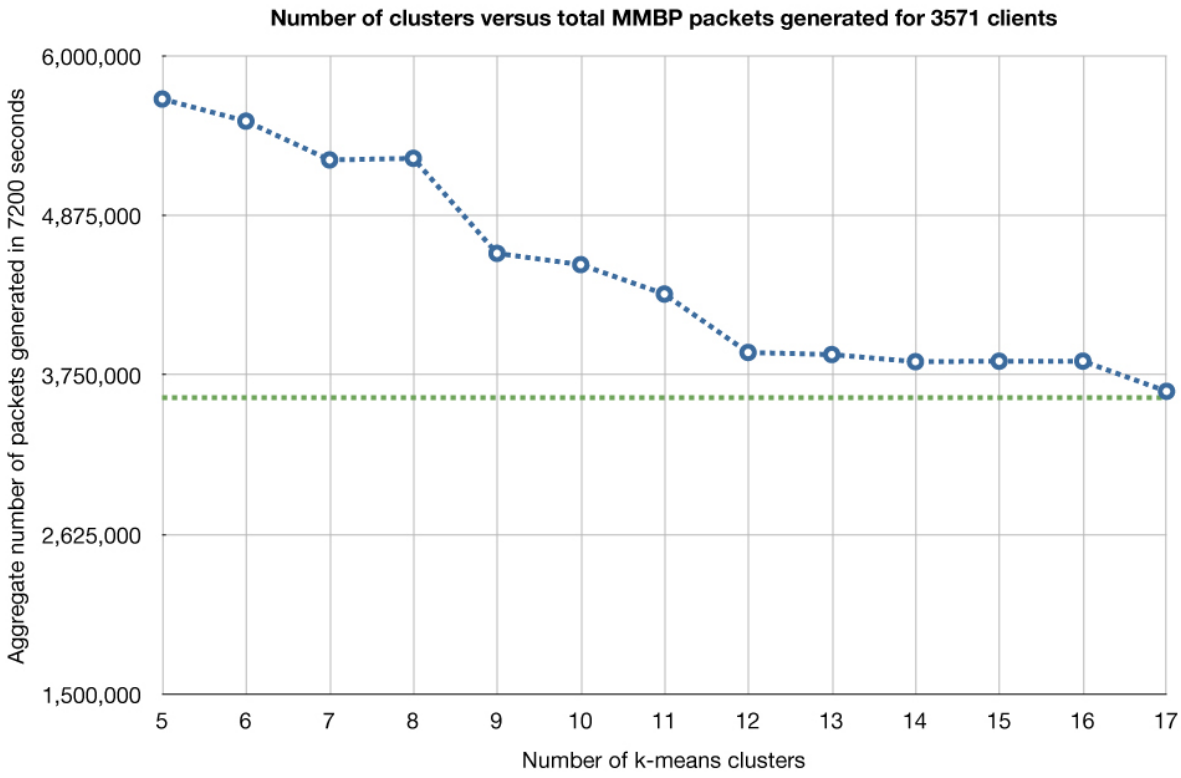

Figure 4.9: Relationship between D-BMAP packets generated and number of clusters

Table 4.3: D-BMAP Packet Arrival Probability per 802.11 Slot

State	Value	**State**	Value	**State**	Value
Idle	0.000000e+00	7	7.236439e-02	14	4.276107e-04
1	1.867082e-03	8	2.335707e-02	15	4.741797e-03
2	1.309437e-03	9	1.207224e-02	16	3.505967e-05
3	2.442150e-03	10	3.079209e-03		
4	7.916290e-04	11	3.411223e-02		
5	5.884292e-03	12	1.813820e-04		
6	3.777329e-03	13	7.602008e-03		

4.6.2.1 Statistical Analysis

Graphical plots of the arrival process can often be misleading, so the arrival process was also analysed statistically. Again, the data was scaled into four resolutions - 1, 0.1, 0.01 and 0.001 seconds. Synthetic

Table 4.4: Trace : statistics for 4401638 packets

Scale	Mean	Deviation	Skewness	Kurtosis
0.001	0.61	1.08	2.86	13.04
0.01	6.11	5.75	1.15	1.49
0.1	61.13	46.09	0.71	-0.19
1	611.34	403.46	0.65	-0.39

Table 4.5: D-BMAP : statistics for 4355831 packets

Scale	Mean	Deviation	Skewness	Kurtosis
0.001	0.604977	0.80195	1.515	11.988
0.01	6.04977	3.11116	0.789736	1.50727
0.1	60.4977	19.7537	0.720158	0.796623
1	604.977	156.335	0.651929	0.836164

Table 4.6: H-MMPP : statistics for 4406435 packets

Scale	Mean	Deviation	Skewness	Kurtosis
0.001	0.612005	0.794455	1.33684	1.89006
0.01	6.12005	2.83978	0.575749	0.488126
0.1	61.2005	16.0272	0.457994	0.698682
1	612.005	141.862	0.448577	0.901017

packet traces were generated using the D-BMAP and H-MMPP models for the same number of clients and time as the Dartmouth traces. The number of packet arrivals in each discrete time unit was recorded to form the sample. The first four statistical moments - mean, variance, skewness and kurtosis - are measured for each of the samples and recorded in Table 4.4, Table 4.5 and Table 4.6.

The measured statistics suggest that both the D-BMAP and H-MMPP models yield less variance than the traces at all scales. For the D-BMAP model, the loss of variance is a result of the clustering

process, which averages a significant portion of the flows to achieve a state reduction. The slightly lower mean values found in the D-BMAP data is due to the smaller number of packets that were generated in the time frame, which is also likely to be due to averaging taking place in the clustering phase of parameter estimation. The mean number of arrivals for the D-BMAP and H-MMPP are acceptably close to the traces. However, this is not the case for the remaining three statistical moments, which suggests that the distributions are not the same.

4.6.2.2 Long-range Dependency Analysis

The Dartmouth traces exhibited an extremely low degree of LRD at both the flow and packet level. Ethernet LAN traffic has been shown to yield typical Hurst values between 0.7 and 0.9 for both flow and packet inter-arrival [44, 67]. By contrast, the Dartmouth traces measured 0.62 for flows and 0.6 for packets - just over the lower bound for LRD. Possible reasons for this low level of LRD might be one or more of the following:

1. WLAN usage and hardware differs from LAN hardware, which could greatly affect the LRD in the traffic. However, this is unlikely since in a comprehensive study of LRD in wireless traffic, Yu *et al* [67] measure LRD values comparable to those from LANs.

2. Research by Cao *et al* [17] shows that as the network load increases, so the packet arrival process tends towards a Poisson process. Since the Dartmouth traces contain a similar load (packet arrivals per time unit) to the traces used by Muscariello [44], it is also unlikely that this is the case.

3. It may be directly related to the fact that traffic from several smaller distinct access points was merged by the RF sniffer, which was used to capture the trace data.

In this work LRD is measured using the inter-arrival time (in milliseconds) of packets. All simulations were run thirty times and a confidence interval was calculated. As expected, the D-BMAP model generated packets with a measured mean Hurst value of 0.549479, which is well below the 0.6 threshold for LRD. More surprisingly, however, the H-MMPP model produced packets with an even lower measured Hurst value of 0.490380. A summary of values is contained in Table 4.7.

Table 4.7: Packet-level Hurst Measurements

Model	Mean Hurst	95% Confidence
Trace	0.609924	Not applicable
D-BMAP	0.549479	±0.000104
H-MMPP	0.490380	±0.005809

4.6.3 Conclusion

The D-BMAP model was conceived to model a single client's traffic. By contrast, the H-MMPP model was designed to reproduce traffic at the backbone level. Although that H-MMPP model provided a superior fit for *aggregate traffic*, the test bed experiments require a *client* level workload model. The method used to calculate the client session arrival rate for the H-MMPP model did not operate as effectively as predicted. The resultant single client traffic was far too bursty and, when multiple client's traffic was merged together, the modified H-MMPP did not accurately reproduce the aggregate traffic.

All three chosen analytic machine models for DCF are based upon the analysis of a discrete-time Markov chain. For performance modelling it is analytically convenient to have a workload model and a machine model that share the same discrete time unit. The D-BMAP has the added advantage of sharing the 802.11 slot time as a discrete-time unit with the three analytic models for DCF.

For the reasons above the D-BMAP was chosen as the superior of the two models and will be used in the non-saturated test bed experiments. The resultant model is the 17 state D-BMAP with parameters drawn from the Dartmouth traces. All clients share the same transition probability matrix (see the accompanying **dbmap-params.conf** file) and packet arrival probability per state (see Table 4.3). The starting state for an arbitrary client is chosen randomly, weighted by the steady state vector.

CHAPTER 5

The IEEE 802.11g Test Bed

5.1 Objective

Raychaudhuri *et al* [52] suggest that, for a number of reasons, much of the wireless network performance research that is conducted in an academic environment uses simulation over protoyping as a method of verification and benchmarking. It is believed that the primary driving factors behind this affinity towards simulation are the cost of equipment and difficulty of conducting experiments. Table 5.1 provides further insight into advantages and disadvantages of experimentation compared to prototyping.

Table 5.1: Pros and cons of simulation when compared to prototyping

Pros	**Cons**
Lower equipment cost	Requires a simulation model
Ability to simply omit exogenous factors	Some effects are near impossible to model
Experiments are easily repeatable	Compounding assumption effect
Results may be practically obtained for large networks	
System time can be accelerated for faster results	

A concerning aspect of analytic model verification via simulation is the compounding effect of assumptions. In order to build wireless network simulation software one is forced to make a number of assumptions about the model. If one compares analytic performance model results directly with simulation results, the deviation describes *how well the analytic model fits the simulation*. Unless the simulation is a perfect representation of the whole system, prototype measurements provide a more accurate insight into real-world performance.

5.2 Requirements

The lack of analytic performance model verification via prototyping is a void in the research area of wireless networking. Therefore, this dissertation provides a direct comparison between solutions from three popular analytic models for DCF with results from a prototype test bed. One key aspect of this work was the design and assembly of an 802.11 test bed with following criteria:

1. **Affordable** - fits within the Data Network Architecture (DNA) research group's budget for hardware acquisition.

2. **Configurable** - in addition to this research, the test bed was used by Stephen Asherson [13] to measure the performance of two end-to-end security extensions for the Optimised Link State Routing (OLSR) protocol.

3. **Expandable** - although initially the test bed was comprised of a sufficient number of stations to carry out the current research, it is designed in a manner whereby more stations could be added should the need arise.

4. **Portable** - since there is limited space in the computer science laboratory, the test bed is designed to be compact and robust enough to store and transport conveniently.

5. **Repeatable** - all stations are equipped with 802.11 wireless Network Interface Cards (NICs) which adhere closely to the standard. All the parameters used to configure the test bed are documented.

5.3 Design

Figure 5.1 shows the design of the test bed, which is comprised of nine client stations and one central controller. All client stations are connected on two distinct networks using a 100 Mbps Ethernet card and an 802.11g wireless card. The latter is reserved purely for experimentation, while the former is used for the following management functions - booting the client stations, controlling the experiments and collecting results. The central controller is responsible for all of these management functions and is

thus also connected to the wired control network. In addition, a spectrum analyser is used to monitor the wireless channel conditions before and during the experiments.

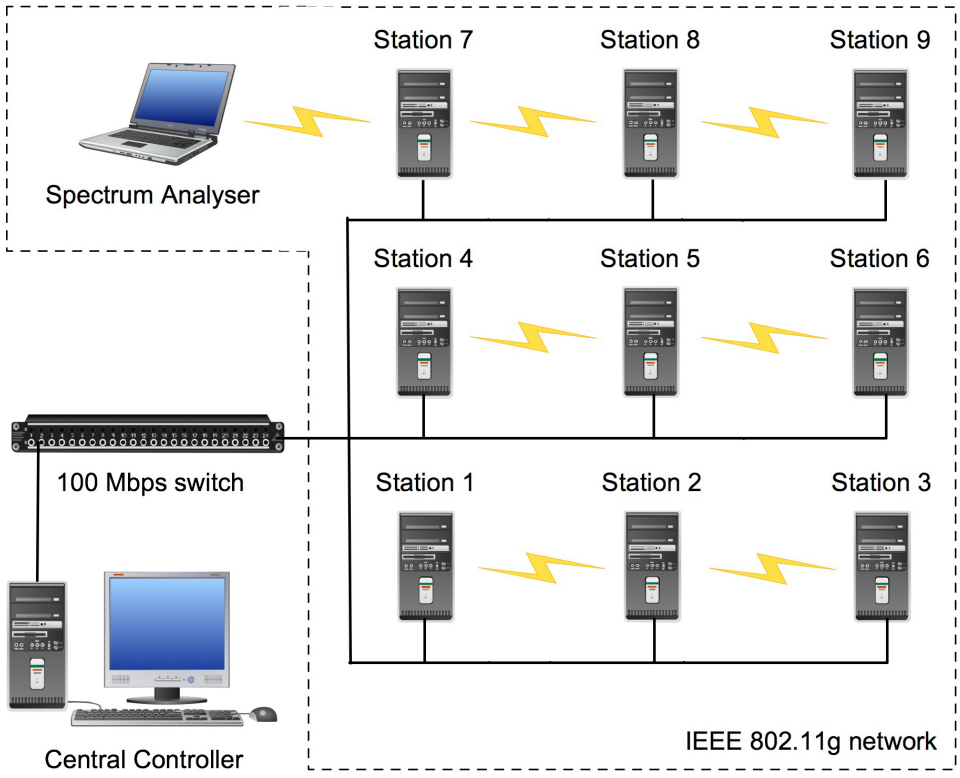

Figure 5.1: The 802.11 test bed design

5.3.1 Base Client Hardware

Three different test bed designs were initially proposed:

1. **Linksys WRT54GL routers** - these routers are significantly cheaper than other options, as they have very limited resources and debugging facilities (no screen or keyboard). The fixed and closed source Broadcom wireless chipset limits the flexibility of the router, despite the open source firmware for the router.

2. **Mini-ITX computers** - although the most expensive option, Mini-ITX computers allow for the greatest level of flexibility. The convenience of choosing a wireless card is advantageous, since open source support varies significantly between manufacturers.

3. **Refurbished thin clients** - the UCT computer science department possessed a number of unused Igel thin clients, which were originally from an undergraduate laboratory. As standalone units they had sufficient resources, with the exception of an 802.11 PCI card.

Initially, the third option was chosen. SMC wireless cards, with the Atheros AR5212 802.11g chipset, proved the most affordable and flexible option for the test bed. However, it was not immediately evident that the PCI version between the two devices differed. Modern cards are built to PCI version 2.2 specification and very few have PCI 2.1 support. The SMCWPCIT-G model did not support PCI 2.1 used by the Igel thin clients and the cards were not recognised by the motherboard.

Subsequently, it was decided that Mini-ITX computers be used in place of the thin clients. Mini-ITX is a reduced size motherboard standard, which makes overall size of the computer significantly smaller. Technical specifications for the Mini-ITX stations may be found in Appendix C.

5.3.2 The Antenna Chain

Another major design consideration for the test bed was the antenna chain. Firstly, the antenna was lifted at least 50cm from the Mini-ITX computer to prevent the metal chassis from interfering with the radio propagation. Secondly, a fixed attenuator was coupled between the card and the antenna to forcibly reduce the signal strength with the goal of artificially creating a multi-hop environment. Although this document does not concern multihop networks, it was used by other researchers in the DNA Research Group.

One of several reasons for the choice of the SMCWPCIT-G wireless cards was that they were each shipped with a detachable external dipole antenna, which was connected to the card using the Reverse Polarity SMA (RP-SMA) connector standard. Figure 5.2 shows how the SMA attenuator was fitted to the antenna chain.

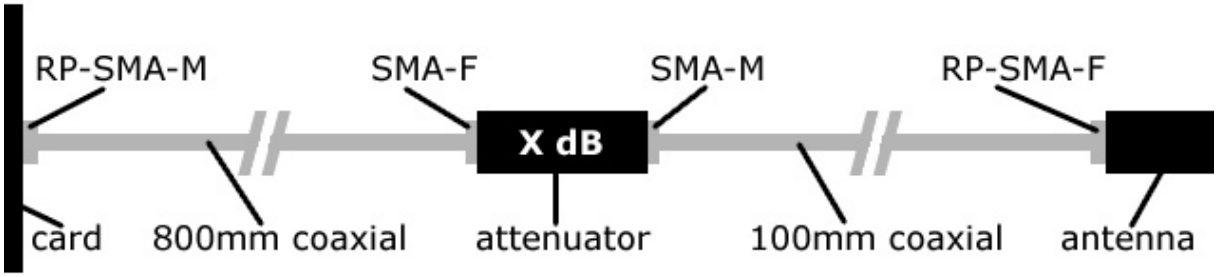

Figure 5.2: Antenna chain with attenuator

5.4 Methodology

5.4.1 Selecting the Attenuation Value for a Multi-hop Test Bed

Refer to Figure 5.3 and consider two 802.11 radios. A radio signal leaves the transmitter's NIC and, as it passes through the radio chain, various electrical components either provide gain or attenuate the signal. In the diagram gains and losses are marked with plus and minus signs respectively.

In order to assess whether a signal is strong enough to support a particular modulation scheme, one calculates the Received Signal Strength Indicator (RSSI) at the receiver and compares it to the threshold provided by the manufacturer. The RSSI value is calculated by taking the net power transmitted, subtracting any losses and adding any gains (see Equation 5.1).

$$RSSI_{dB} = TxPower_{dBm} + Gains_{dB} - Losses_{dB} \quad (5.1)$$

$$Gains_{dB} = 2 \times AntennaGain_{dBi} \quad (5.2)$$

$$Losses_{dB} = FSL_{dB} + 2 \times (Attenuator_{dBi} + 4 \times Connector_{dB} + Cable_{dB}) \quad (5.3)$$

The $TxPower_{dBm}$ value is simply the Decibel notation of output power of the 802.11 radio, relative to 1 milliwatt. The SMCWPCIT-G can transmit in the integer range 0 to 15 dBm. The gain in the

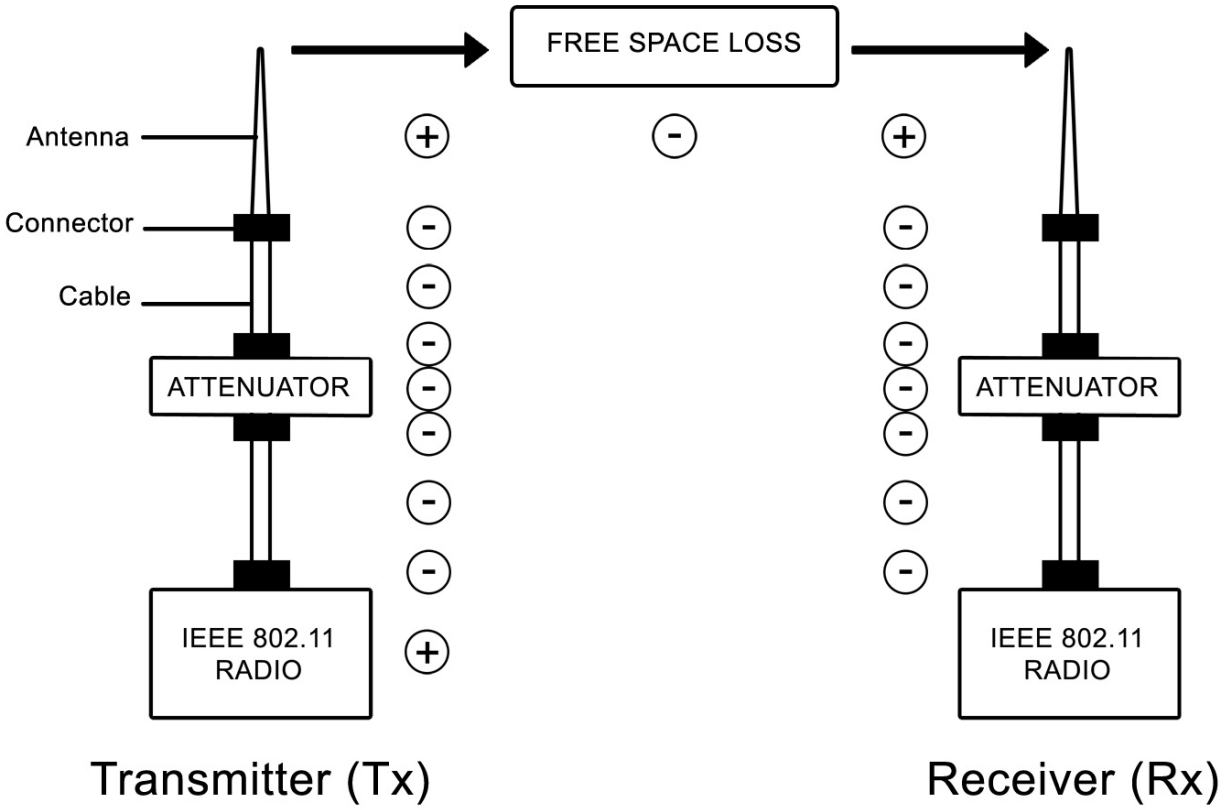

Figure 5.3: Calculating the link bugdet for two IEEE 802.11 radios

system comes in the form of two 2 dBi antennas and is shown by Equation 5.2. Free space, cables, connectors and attenuators reduce the strength of the signal according to Equation 5.3.

Free space loss is the reduction in signal strength that occurs purely as a function of the distance separating the transmitter and receiver. It is shown as Equation 5.4 and is dependent on the centre frequency (f_{MHz}) of the band and the distance (S_{km}). Additionally, a multiplier (k) is used to represent how readily the signal propagates though the space. At standard room temperature $k = 2$.

$$FSL_{dBm} = 32.5 + 20\log(f_{MHz}) + k \times 10\log(S_{km}) \tag{5.4}$$

Using Equations 5.1 to 5.4 it is possible to calculate the signal strength at the receiver as a function of three independent variables - the transmit power, attenuation value and distance between stations.

The available fixed attenuators are limited to the following values - 10, 12, 15, 20 and 30 dB. One can couple several attenuators but this is a prohibitively expensive strategy.

The SMCWPCIT-G specification sheet indicated that a 1 Mbps mode requires a -86 dBm RSSI value at the receiver. Using a 20 dB fixed attenuator at both stations with the lowest available transmission power, 0 dBm, the test bed required a spacing of around 10 meters between client stations. However, with a 30 dB attenuator it was possible to achieve a fade margin of 1.5 m, permitting stations to be spaced at 1 m. However, at high OFDM rates 30 dB attenuation was too strong, as there was no practical spacing which allowed for a receive strength above the threshold.

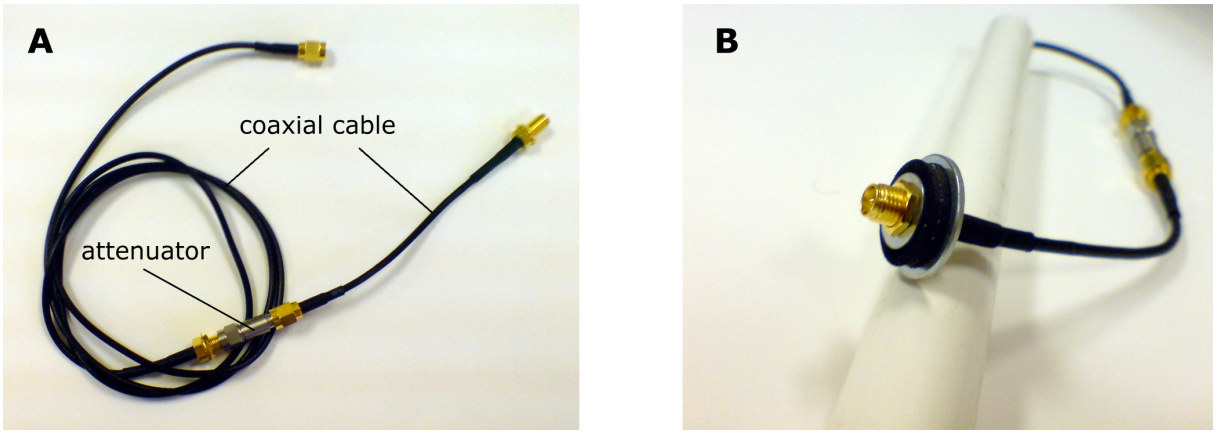

Figure 5.4: The antenna chain (a) without enclosure and (b) partially inserted in the PVC tubing

5.4.2 Client Station Hardware Assembly

The first assembly task was to cut two antenna shafts per station from 30.5mm medium grade plastic tubing. This type of tubing is normally used for household electrical wiring and is available from most hardware stores. The second antenna shaft is used to reinforce the first to prevent too much bending from taking place during experimentation. The length of the shaft was chosen as 600mm, so as to distance the antenna sufficiently from the chassis while providing a small amount of slack for the cabling. After the antenna shafts were cut the antenna chain (cabling, connectors and attenuator) is

threaded through the tubing. The completed antenna chain and shaft is shown in Figure 5.4.

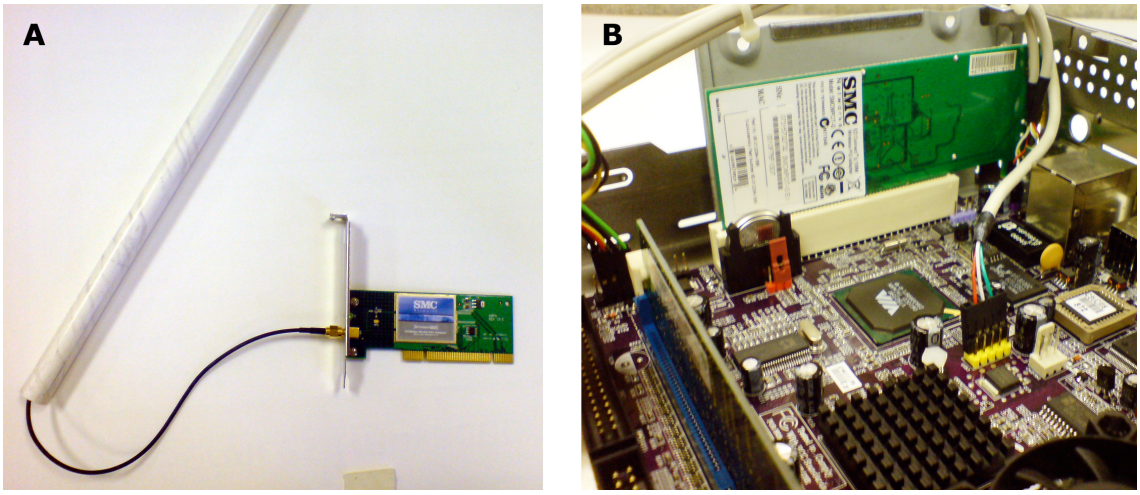

Figure 5.5: The (a) completed antenna with card and (b) card inserted into Mini-ITX client

Once the antenna chain and shaft were built, the wireless card was inserted into the Mini-ITX client (see Figure 5.5). Note that the PCI metal bracket was removed from the wireless card prior to insertion, as the PCI slot in the Mini-ITX client is half-size and cannot accommodate a full size bracket. The loss of the bracket makes the positioning of the cards less secure, so care was taken to ensure that the cards are seated firmly in place and stations are transported carefully.

The final task was to arrange the stations into the final test network. For this dissertation the physical positioning had no significant influence over results, as full-connectivity between stations was a network requirement. The 3x3 grid layout of the client stations was chosen pragmatically.

5.4.3 Software Assembly

All of the client stations did not have disk drives installed and therefore booted directly off the wired network using the Pre-boot Execution Environment (PXE) protocol, which makes use of the Trivial File Transfer Protocol (TFTP) to transfer operating system images to clients. The Linux operating system into which they booted was stored on the central controller and all changes to each client's

Figure 5.6: (a) measuring the distance between stations and (b) the completed test bed

file system were lost when the stations were power cycled. Therefore, to preserve information across reboots, each client station mounted a unique mutable directory via Network File System (NFS) off the central controller. In summary, the following boot process was followed:

1. The client issues a Dynamic Host Configuration Protocol (DHCP) request on the wired network.
2. The controller responds with PXE and network information based on the client's MAC address.
3. The client requests the operating system images via TFTP from the central controller.
4. The client loads the kernel image into memory and boots into the operating system.
5. On booting the kernel extracts the initramfs into memory and binds it to the system root.
6. The init script in the initramfs configures the client and mounts the mutable share.

5.4.3.1 Configuring the central controller for network booting

All system development and experiment control is managed by the central controller. While it is, in theory, possible to use any Unix or Linux based operating system for development, *Gentoo Linux* [3] was chosen for this project. The primary reason for this choice was that the *Portage* package management

system, in conjuction with the *Gentoo Embedded* [5] project, provided an useful platform for developing very small initramfs images. Once the Gentoo base system was prepared, the following system packages were installed and configured on the central controller:

- A DHCP server (**dhcpd**) that issued each station a fixed IP and PXE information.
- A TFTP server (**tftp-hpa**) that shared kernel and initramfs images over the network.
- An NFS server (**nfs-utils**) that offered a unique mutable mount for each client station.

PXELINUX [12] provided instructions on how to prepare a PXE network boot environment using the DHCP and TFTP daemon services in conjunction with the distributed **pxelinux.0** image and an appropriate configuration script.

5.4.3.2 Preparing the development environment and building the initramfs image

uClibc and *uClibc++* are lightweight alternatives to *glibc* and *libstdc++*, the default GNU C and C++ libraries used by modern Linux distributions. Linking applications against these alternative libraries results in significantly smaller application software. The *Gentoo Embedded* project simplifies this task by providing a **Stage 3 embedded tarball** with system and development libraries pre-linked against *uClibc*. The development environment is comprised of the following three nested root file systems (RootFS). Switching between the three environments is achieved using the **chroot** application.

- **System RootFS** - The default Gentoo Installation on the central controller.
- **Development RootFS** - The Gentoo Embedded installation, linked against uClibc and libstd++, located in the **/opt/embedded-toolchain** directory relative to the System RootFS.
- **Embedded RootFS** - The client root file system, linked against uClibc and uClibc++, located int the **/opt/meshnet-rootfs** directory relative to the Development RootFS.

The initramfs was developed using a package called **baselayout-lite**[1] which is, essentially, a lightweight Linux base file structure containing only essential files and directories. Portage was instructed to bind the baselayout-lite package to a custom directory which formed the base of the Embedded RootFS.

Following this, the Embedded RootFS was populated with the necessary system binaries, user applications and configuration files. Although this approach was more time consuming than using a prebuilt distribution, like *Damn Small Linux* [1], it allows complete control over the applications that are installed. The advantage of this level of control is that it results in a much smaller RootFS containing only applications and daemon services configured specifically not to interfere with the experiments. The process of installation and configuration of this system is beyond the scope of this document.

5.4.3.3 Writing the client initialisation script

After the kernel has booted and the initramfs has been extracted to a RAM file system and bound to root, the kernel executes an initialisation (init) script, which is usually named *linuxrc* or *init* and located in the system root. The most straightforward method of initialisation is to symbolically link this script to **/bin/busybox**. After performing some of its own initialisation routines, this binary calls a custom initialisation script in **/etc/init.d** which performs the following:

- Starts network services - DHCP client, portmap, SSH daemon, mounting NFS share.
- Manages the wireless interface - loading kernel modules and configuring the network interface.
- Sets up a fixed wireless ARP table - prevents ARP requests from interfering with experiments.
- Configures experiments - manages kernel panics caused by the MadWiFi [6] drivers.
- Starts custom applications - sink and listener daemons for experiment control.

[1] At the time of writing baselayout-lite has been removed from the Gentoo Portage package management system and replaced with baselayout2. Refer to the Gentoo Embedded project for further information about changes to the package.

5.4.3.4 Compiling the client kernel

The majority of time spent compiling a Linux kernel [58] involves choosing the correct kernel configuration for the target hardware. Refer to the Gentoo Handbook [4] for more information about this process. It is advised that the kernel be compiled within the Development RootFS to keep them separate from the System RootFS kernel. The following kernel options were set to enable support for uClibc and initial RAM filesystems (a requirement for PXE booting):

```
General Setup --->
 Configure standard kernel features (for small systems)
  [*] Enable 16-bit UID calls
Device Drivers -->
 Block devices
  [*] Loopback device support
  [*] RAM disk support
  ...
  [*] Inital RAM file system and RAM disk
```

5.4.3.5 Packaging the initramfs and kernel images

In addition to building the kernel image Linux kernel compilation also produces kernel modules. Although one may configure almost all the functionality to be built-in to the kernel image, some third-party drivers, such as the ones provided by the MadWifi project, have to be compiled as modules. Before the initramfs is created, the modules existing in the **/lib/modules** directory of the System RootFS must first be copied to the **/lib/modules** directory of the Embedded RootFS.

At this stage all custom software, including the applications which control the test bed experiments (**meshnet-tools**), was copied to the Embedded RootFS. The initramfs was be created by taking a snapshot of this directory and converting it to compressed cpio binary archive. This was achieved by issuing the following command in the **/opt/meshnet-rootfs** directory of the Development RootFS.

The resulting **initramfs.gz** file was the initial RAM fileystem.

```
$> find . | cpio -o -H newc | gzip > ../initramfs.gz
```

The final task involved copying the initramfs and kernel images to the TFTP share. For convenience the initramfs and kernel images were renamed to **initramfs-pxe** and **kernel-pxe** respectively. Prebuilt versions of these images are available for download from the project website [56].

5.4.4 Spectrum Analyser Station

Prior to experimentation the 2.4 GHz ISM band was monitored using a spectrum analyser to ensure that there was no exogenous source of interference that might have negatively affected the performance of the 802.11 channel access control mechanism. To achieve this a low-cost Wi-Spy 2.4 GHz USB spectrum analyser was used in conjunction with the freely-available EaKiu [2] driver software.

5.5 MadWifi Interface Settings

The MadWifi driver set creates a new interface (denoted wifiX, where X is an integer) for each wireless NIC in the system. Using the **wlanconfig** application it is possible to create virtual interfaces that make use of this actual interface. The MadWifi drivers offer a number of virtual interface types including ad hoc, access point, monitor (promiscuous mode) and ahdemo. For the test bed the ahdemo mode was chosen, as it prevents ad hoc association problems (BSSID partitioning) and removes beacon frames, both of which negatively affect the performance of the channel access control protocol.

Since there are a number of different parameters to configure an 802.11 network interface, one of the crucial aspects of the test bed design is understanding how to correctly configure the hardware, so that the measured results will be comparable to the analytic results. Appendix F provides a lists of the important configuration parameters, how they relate to the 802.11 standard and the commandline instruction used to set them (under the assumption that the MadWifi drivers are being used).

CHAPTER 6

Experimentation

6.1 Objective and Assumptions

The broad goal of experimentation is to empirically evaluate how, for 802.11 Internet access networks, the DCF channel access control mechanism scales in artificial saturation conditions versus more realistic workload conditions. The following assumptions are made about the 802.11 machine model:

1. A fully-connected network with a single hop between all stations.
2. Perfect channel conditions - no interference from external sources.
3. All stations share a common 2.4 GHz 802.11g ERP-OFDM PHY, configured in the following way:
   (a) Pure mode operation (no 802.11b protection) with a $9\mu s$ slot time.
   (b) No proprietary extensions such as turbo mode, compression or frame bursting.
   (c) Fixed rates of 54 Mbps and 24 Mbps for data and control frames respectively.
4. DCF channel access - both basic and RTS/CTS access modes will be tested.
5. The fragmentation mechanism is disabled.
6. Only RTS, CTS, DATA and ACK frames (no beacon, association or other management frames).
7. A transmission from a single client station to the nominated network sink has the effect equivalent to a transmission occurring in the opposite direction. Therefore, queuing at the sink's network layer has no role in determining the network performance. Performance is determined entirely by the operation of the channel access control protocol and measured by normalised throughput.

6.2 Experiment Software

The software used to control the test bed experiments comprises of three applications:

- **meshnet-controller** - This is the application responsible for controlling an experiment and is executed on the central controller. It takes a single configuration file as an argument and, based on the content of the file, the application configures the test bed and conducts experiments.
- **meshnet-listener** - This application runs permanently as a daemon on the client stations. It listens for incoming requests over the wired network from the meshnet-controller application.
- **meshnet-sink** - Although it is run as a daemon on all client stations, this application is only used by the single station nominated as the network sink. The sole purpose of this application is to absorb and acknowledge traffic sent by client stations over the wireless network.

All experiments are instigated from the central controller. If required, the central controller transfers all workload configuration files to the stations in the network before the experiments begin. Then, the central controller sends configuration instructions to the sink and client stations to prepare them for the upcoming experiments. In the central controller configuration file one specifies a range of network sizes to test. The central controller treats each size as an experiment, repeating each experiment for a fixed number of runs in order to obtain numerical accuracy. At the beginning of each run a new subset of the available stations is chosen randomly. Each of the selected stations is then instructed to transmit packets to the sink according to some workload model (either saturation or D-BMAP) for a fixed period of time. After this time has elapsed, each station responds with the number of packets that were successfully transmitted. At the end of the run the central controller calculates the station average number of packets transmitted per station over the experiment duration and saves the result. Analysis and formatting occurs after all the experiment data has been gathered.

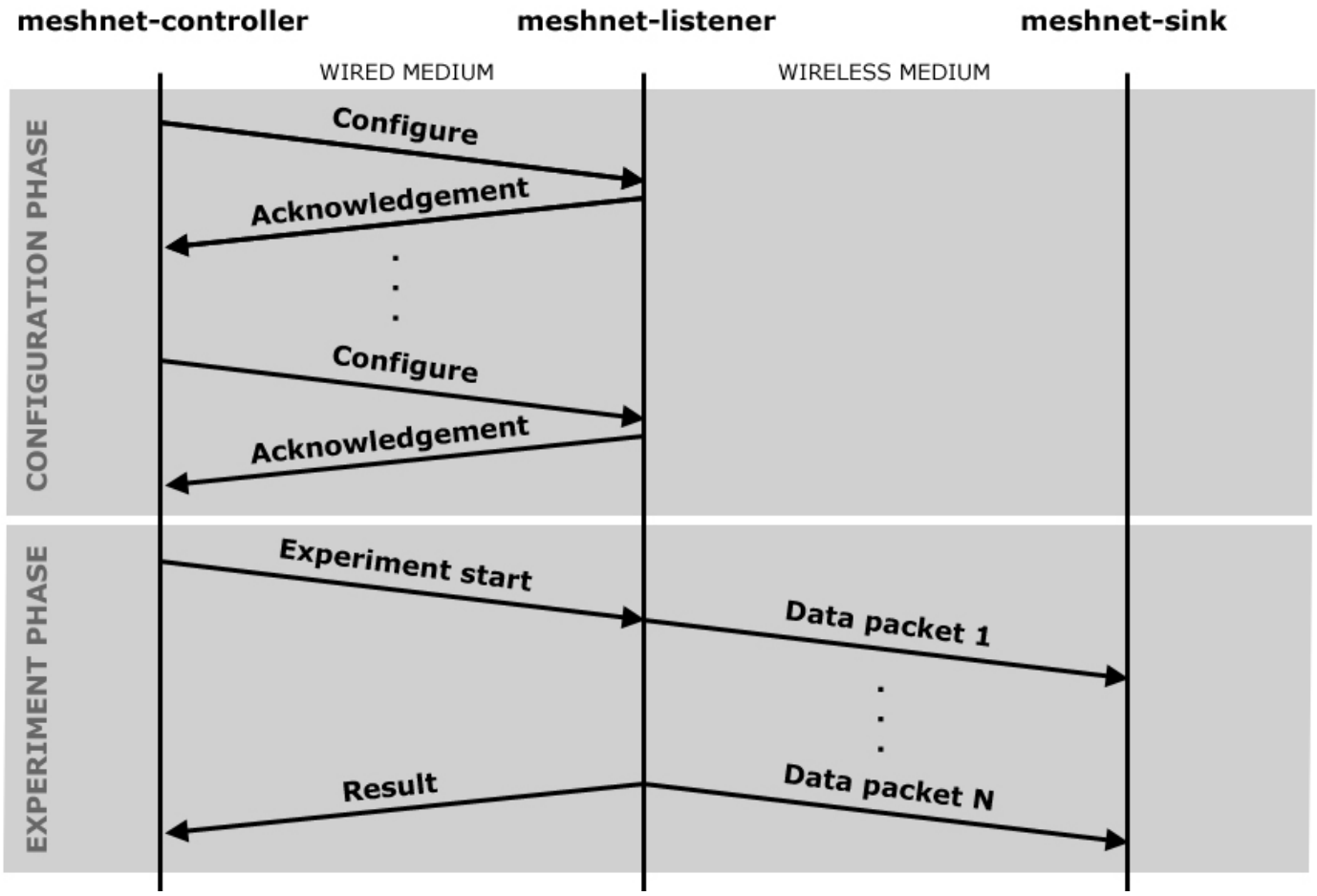

Figure 6.1: The network protocol used by the **meshnet-tools** software

6.3 Selecting an Experiment Location

Of the three chosen analytic models for DCF, only Szczypiorski's [57] accounts for interference. To ensure that solutions amongst analytic models and measurements from the test bed are comparable the Bit Error Rate (BER) is set to zero. Unfortunately, this is a theoretical system state that one cannot recreate perfectly in a test bed environment. In order for the experiment results to be comparable with the analytic results, the effect of external interference was reduced to as little as possible.

One method to reduce exogenous radio interference is through the use of an anechoic chamber. Neither the DNA Research Group or the University of Cape Town has access to such a facility. Therefore,

it was decided that the test bed be moved to two remote locations. The first location was in a guesthouse on farm in Franschhoek and the second was in the basement of a house in Hout Bay.

6.4 Interference Tests

To assess the channel conditions in Franschhoek two preliminary tests were run - a *channel analysis* using the Wi-Spy spectrum analyser and a *frame analysis* using the one of the wireless cards in the client stations in conjunction with a popular network sniffer tool. Both tests were run concurrently for 30 minutes within 2 meters of all client stations.

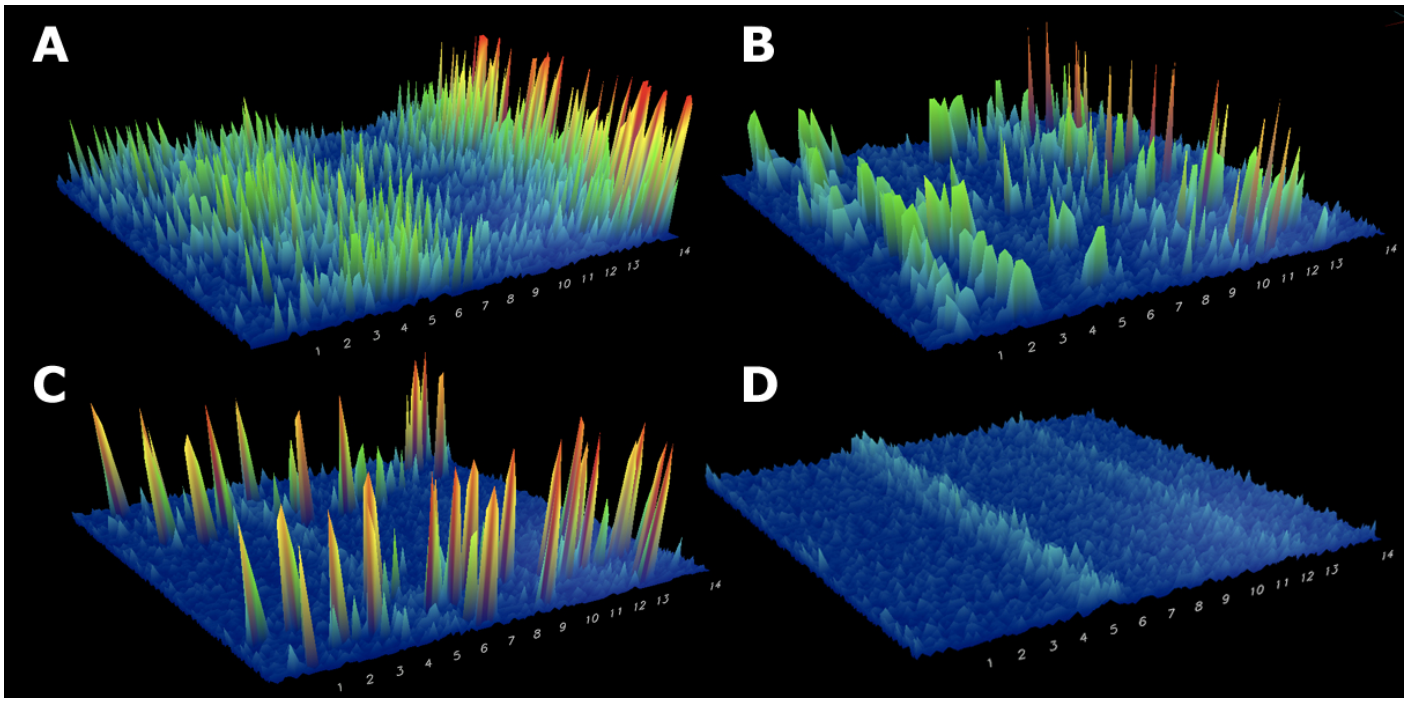

Figure 6.2: WiSpy traces - (A) microwave oven, (B) 802.11, (C) 802.15.1, and (D) clear channel

6.4.1 Power Analysis

The Wi-Spy spectrum analyser is capable of detecting power levels in the range -97 dBm to -50.5 dBm at a 1.5 dBm. It sweeps the across the 2400 to 2482 MHz band in 120 ms, sampling the received power

level at every 1 MHz interval [7]. A newer version of the Wi-Spy (2.4x) is also available, which samples at a superior resolution for a wider range of power levels (-110 dBm to -6.5 dBm).

Figure 6.2 illustrates graphically 30 seconds worth of sample power levels recorded by Eiaku under controlled conditions for a microwave oven, 802.11 network and Bluetooth device. It is interesting to contrast the signal pattern of the FHSS modulation technique used by 802.1514 in (C) with the OFDM (about channel 1) and HR/DSSS (around channel 11) modulation used by 802.11 in (B).

Figure 6.2 also shows a 30 second snapshot of the channel conditions in Franschhoek. What is immediately evident is the presence of two separate and faint (less than -87 dBm) signals on channel 6 and 11. Based on this observation the only other non-overlapping 802.11 channel, centred at 2.412 GHz (channel 1), was used to conduct the experiments. Similar channel conditions were measured in Hout Bay. However, a single access point was observed broadcasting frames on channel 11. To avoid channel overlap with the access point, experimentation was conducted on channel 1.

6.4.2 Frame Analysis

Although the channel analysis provided an accurate readout of signal strength, it cannot detect and display faint 802.11 frames. Very low power 802.11 frames have the potential to disrupt experiments. Consider a third-party 802.11 station that transmits an RTS frame at 1 Mbps which is successfully received by one of the test bed stations. The receiving station is forced to defer access based on the DCF protocol for the duration in the RTS frame header. This consumes a large amount of channel time which could otherwise be used for data transmission.

Therefore, in addition to the channel analysis a frame analysis is performed. To begin the process a new MadWiFi *monitor* virtual interface is created on a test bed station. This type of wireless interface supports Radio Frequency Monitor *RFMON* mode, which allows one to capture wireless frames, regardless of whether or not association has taken place. After some configuration, the **tcpdump** [8] application is used to capture all frames received. The corresponding shell script follows:

```
$> wlanconfig ath1 create wlandev wifi0 wlanmode monitor
```

```
$> iwconfig ath1 channel 1
$> ifconfig ath1 up
$> tcpdump -i ath1 -vv
```

Regular 802.11 stations broadcast management frames periodically at the control rate. In Franschhoek, for the entire duration of the test no frames were captured by tcpdump on channel 1. The fact that no frames were received throughout the entire sampling period makes it an ideal location for the experiments. In Hout Bay short probe requests were received periodically on channel 1 from a neighbouring 802.11 access point on channel 11. Probe requests are used by access points to periodically gather information about neighbouring 802.11 stations. Their short length and long inter-arrival time means that the overall effect on network performance is negligible. Despite the existence of these probe requests, Hout Bay was chosen as an appropriate second location for experimentation.

Table 6.1: Summary of the test bed configuration

Configuration	**Option**	**Value**
PHY Layer	Mode	802.11g
	802.11b protection	*disabled*
	802.11b rate support	*disabled*
	Slot time	$9\mu s$
	SIFS	$10\mu s$
	CW_{min}	16 slots
	CW_{max}	1024 slots
	Short retry count (RTS/CTS access)	7
	Long retry count (basic access)	5
	Short preamble	*not required*
MAC layer	Data rate	54 Mbps fixed
	Basic rate	24 Mbps fixed

6.5 Configuring the Fixed Machine Model

The Atheros AR5212 chipset supports the modulation schemes introduced by both the 802.11b and 802.11g standard revisions. It does not support the 2.4 GHz IR and FHSS modulation schemes used in the legacy 802.11 standard or the 5 GHz 802.11a OFDM modulation scheme. The number of channels is limited by the local government or telecommunications regulator. Based on the interference tests channel 1 was chosen for experimentation. The basic and data transmission rates were fixed at the maximum permissible speeds of 24 Mbps and 54 Mbps respectively.

In addition, the test bed was configured to operate in 802.11g *pure mode*, which does not support older 802.11b stations. The slot time was reduced from $20\mu s$ to $9\mu s$ and the 802.11b protection mode was disabled. The minimum and maximum contention windows remained at 32 and 1024 respectively, despite the reduced slot time. Since no DSSS modulated data was sent, the short preamble was not required or used. Table 6.1 provides a configuration summary.

Table 6.2: Parameters for all four experiments

Parameter	**Experiment 1**	**Experiment 2**	**Experiment 3**	**Experiment 4**
Workload (arrival)	Saturation		D-BMAP workload model	
Workload (length)	Fixed at 1000 Bytes		Clustered log-normal	
Access mode	Basic	RTS/CTS	Basic	RTS/CTS
Network scale	Number of contending stations varied between 1 and 8			
Runs	30 samples		15 samples	
Run duration	60 seconds		300 seconds	
Total duration	**4h00m**	**4h00m**	**10h00m**	**10h00m**

6.6 The Four Experiments

The goal of the first test bed experiment was to measure saturated DCF performance and compare it with analytic solutions for the same experiment. The goal of the second test bed experiment was to measure non-saturated D-BMAP performance on the test bed and compare it with the measured saturation performance. In both cases the experiments are repeated for both basic and RTS/CTS access. Therefore, there are four experiment sets listed as columns in Table 6.2. To assess the scalability of DCF, within each experiment set the number of contending stations is scaled from one to eight, yielding eight experiments per experiment set. Depending on the workload model, each experiment is run a number of times for a particular duration to ensure that results are obtained at a 95% confidence level.

The saturation experiments (Experiment sets 1 and 2) were conducted first at the Franschhoek location. Afterwards, the result were analysed and compared with the analytic models for saturation. Then, the D-BMAP experiments (Experiment sets 3 and 4) were conducted in Hout Bay. The Franschhoek and Hout Bay experiment run times were 8 and 20 hours respectively. The total experiment time was approximately 28 hours plus the time overhead required to switch between runs and experiments.

The experiment software uses the Unix socket API and the UDP transport mechanism. Before being processed by the MAC a UDP, IP and SNAP header of 8, 20 and 8 bytes respectively are appended to every data frame (see Appendix B for further details on framing). Therefore, in the case of saturation, in order to achieve an MPDU size of 1000 bytes the socket-level data frame size is set to 964 Bytes. For the D-BMAP experiments no headers are subtracted, as the packet sizes measured from the tcpdump traces reflected the true packet payload length.

CHAPTER 7

Results

7.1 Experiment Results

Figure 7.1 provides a summary of the results from the two saturation experiments. Each of the data points and their associated 95% confidence intervals are calculated using 30 sample readings of 60 seconds each. An operating system error[1] forced two independent runs from the basic access experiments to return disproportionally large results. These runs were discarded before the statistics were calculated.

As expected, for a small number of stations the medium is underutilised; when contention is low there are lengthy periods of back-off in which no competing stations capture the medium. Normalised aggregate throughput peaks at two and three stations for basic and RTS/CTS access respectively. After these peaks, the cost of collisions outweighs the time advantage brought by the greater probability of earlier medium capture; more stations imply a greater capture probability in an arbitrary idle slot.

The four way handshaking process used by RTS/CTS adds a significant amount of overhead to each data frame. This is clearly indicated by the fact that, for a network containing a single station, basic access yields an aggregate throughput value of 0.4323 whereas RTS/CTS access yields only 0.3419. Just over 9% of overhead is thus added by the additional handshaking process. More interestingly however, as the number of contending stations in the network increases, so this difference gets smaller. Between three and eight stations the measured data suggests that the reduction in overhead is approximately linear. At eight stations the RTS/CTS access mechanism adds around 2% extra overhead than basic access. If one assumes linearity and extrapolates, it turns out that the RTS/CTS mechanism **reduces** overhead by about 8% at sixteen contending stations.

[1]The MadWiFi drivers caused a Linux kernel OooPS in both cases and the remaining runs were unaffected

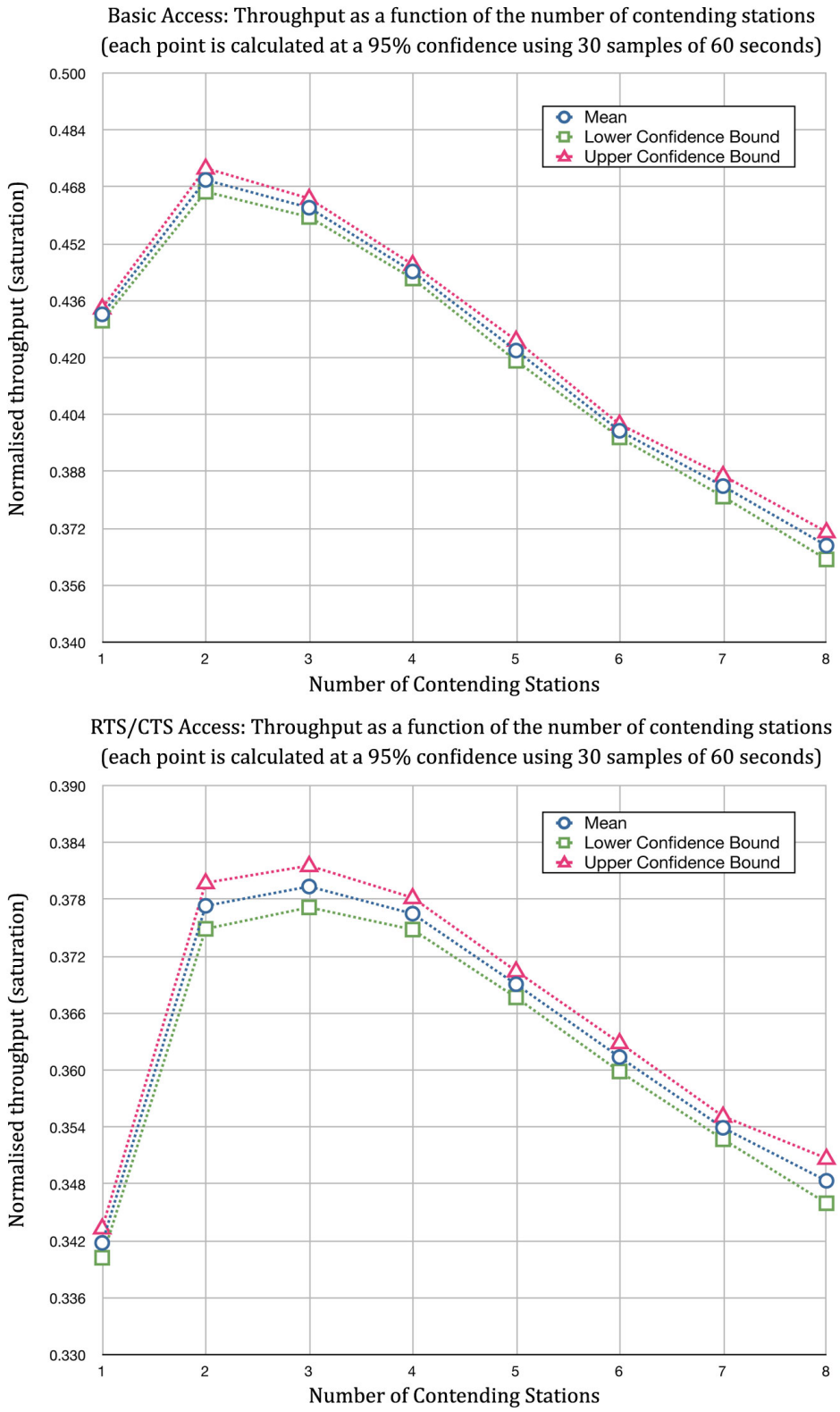

Figure 7.1: Normalised aggregate saturation throughput versus number of contending stations

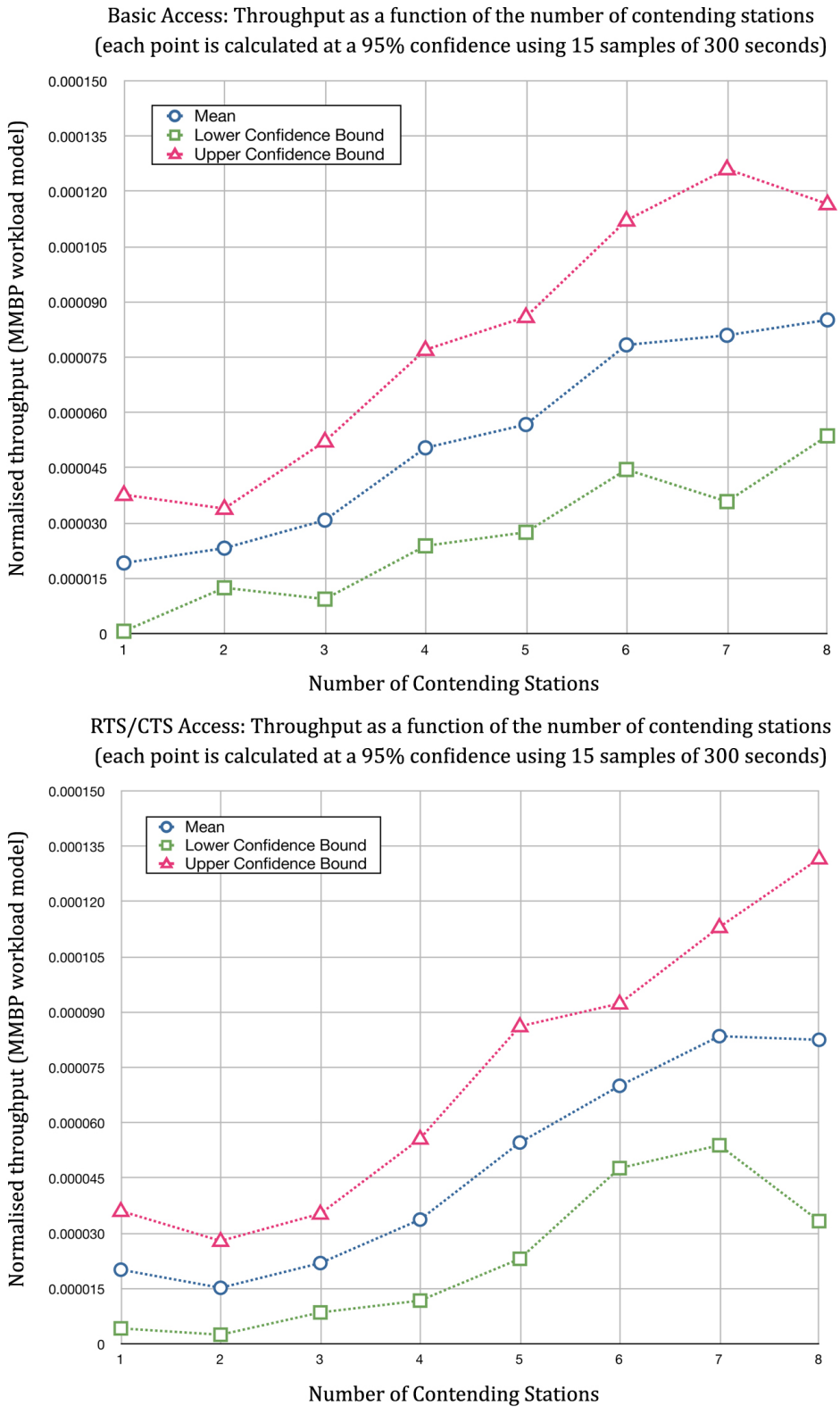

Figure 7.2: Normalised aggregate D-BMAP throughput versus number of contending stations

Figure 7.2 provides a summary of the results from the two D-BMAP experiments. Unlike the saturation experiments, each point's mean and confidence interval were calculated using 15 sample readings of 300 seconds. The experiment time was doubled to accommodate any unexpected convergence period as a result of the D-BMAP model, while the number of runs was halved to maintain the same total experiment duration as the saturation experiments. No errors were reported during the experiments.

Like the saturation results, at the point where eight stations contend for channel access the aggregate throughput in both the basic and RTS/CTS access experiments are approximately equal. For up to five stations, the overhead added by the RTS/CTS access mechanism reduces the aggregate system performance by up to 34.21% relative to basic access. However, like the saturation experiments, as the number of stations contending for access increases, so the performance of the RTS/CTS access mechanism increases relative to the basic access mechanism.

7.2 Analytic Saturation Models versus Experimental Results

To assess how well the analytic models for saturation measured normalised aggregate throughput, the experiment results were compared to numerical results from three widely-accepted analytic models for DCF. Szczypiorski's model was extended to provide a solution for RTS/CTS. Figure 7.3 shows the experiment results and numerical solutions on the same axes, for both basic and RTS/CTS access. The packet length is fixed at 1000 bytes. The experiment data is available for up to eight contending stations, while analytic solutions have no upper bound. To provide an indication of the analytic trend, solutions were calculated up to 16 contending stations.

Bianchi's model offers the greatest overall aggregate performance, as it does not account for both back-off counter suspension and finite retries in DCF. Infinite retry counters ensure channel efficiency at the cost of an unbounded response time. In reality, buffers are finite and there is a maximum tolerable response time, so the retry count on the final back-off stage in DCF is finite. Hence, Wu's model yields an aggregate throughput value lower than Bianchi's when two or more stations contend for channel access. Back-off counter suspension forces collisions to take place only on the first slot of transmission. For the remaining slots channel sensing takes place and collisions are avoided by suspending the back-off

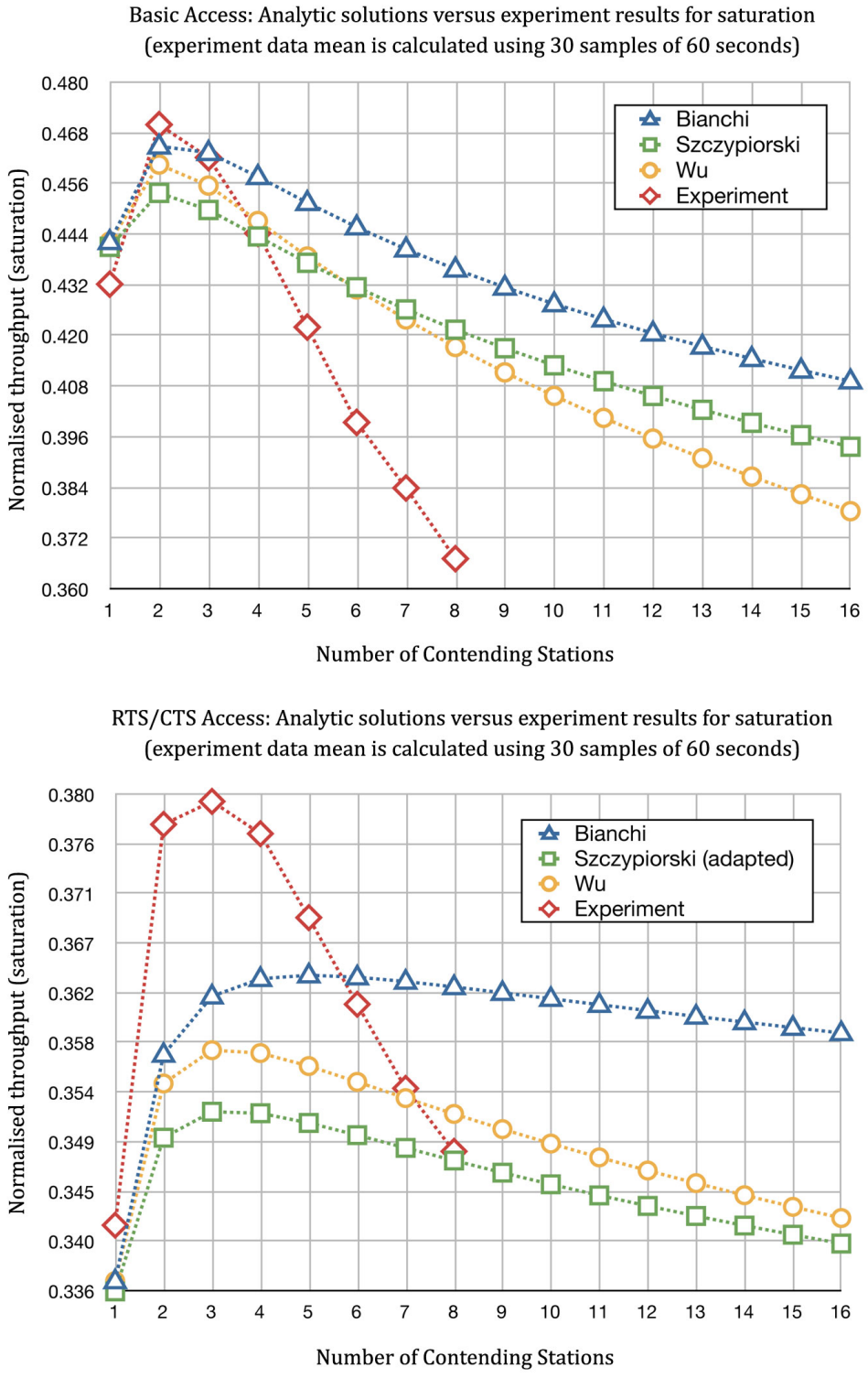

Figure 7.3: Numerical solutions for saturation versus experiment results

counter until the channel is perceived as idle. By factoring back-off counter suspension into an analytic model aggregate throughput is raised, as costly collisions are avoided. Therefore, in situations where the probability and cost of collision is high, Szczypiorski's model calculates aggregate throughput above Wu's model. Conversely, when collision probability is low (for RTS/CTS access or basic access with fewer than six stations) Szczypiorski's model reduces aggregate saturation throughput.

Although for both basic and RTS/CTS access the overall trends between numerical solutions and experiment results are similar, the experiments suggest that aggregate DCF performance degrades faster than predicted by the analytic models. For basic access and very few stations, the results from the experiments and analytic models agree. The opposite is true with RTS/CTS access, where initially the experiments suggest that aggregate throughput is much greater than predicted by the analytic models. Possible causes will be discussed in Chapter 8.

7.3 Saturation experiments versus D-BMAP experiments

The secondary objective of this research was to measure the scalability of the channel access control subject to a real workload model, and compare the results to saturation conditions. As expected, there is a large discrepancy between real-world usage and artificial extreme network conditions. For a real-world traffic model, at the point where eight stations contend using either basic or RTS/CTS access, only 0.023% of the available channel capacity[2] is being used.

For the experiments in which eight stations contended for basic access to the channel, aggregate throughput for the D-BMAP and saturation models were measured as 4.8 Kbps and 19.626 Mbps respectively. Furthermore, there is little difference between the D-BMAP basic and RTS/CTS trends, which suggests that the access method is influential only in DCF networks with a heavy load. The only minor difference between the basic and RTS/CTS D-BMAP experiments is the slope of the curves; for basic access the curve is linear, while for RTS/CTS access the slope changes. However, given the wide confidence intervals it is impossible to infer anything definitive from the data.

[2]Channel capacity is defined here as the normalised aggregate throughput taken from the saturation experiments

CHAPTER 8

Conclusion

8.1 Research Outcomes

The data suggests that, as the number of contending stations increases, so the aggregate throughput measured from the experiments decreases faster than predicted by the analytic models. All three of the analytic models were verified by their respective authors by comparing the numerical results against measurements taken from a DCF simulator of sorts. Therefore, the difference between the analytic results and the experiments suggests that some property of the real-world test bed is not accounted for by simulation. Possible reasons for this discrepancy are the following:

1. At the application layer packets of a predetermined size are written to a UDP socket according to a saturation model. However, the Linux kernel is responsible for buffering and managing these packets in a manner which benefits the typical user. Such a strategy would impose a small amount of delay between a portion of the packets before they are forwarded to the MAC layer for service. This lowers the performance of the test bed relative to analytic models and simulation.

2. In order be to functional, simulators are forced make a number of assumptions about the physical layer, channel conditions and wireless medium. Coupled with varying degrees of abstraction and the large number of parameters associated with wireless networking, this results in large differences amongst popular implementations. Cavin *et al* [18] researched the accuracy of MANET simulators and concluded that, even for simple scenarios, results amongst simulators diverged significantly.

As expected, the aggregate throughput measured in the D-BMAP experiments was significantly lower than the saturation experiments. This was caused by medium under-utilisation, which stems

from having (i) idle clients, (ii) inter-arrival times between transmitted data frames at each client, and (iii) varying sized data frames. A large proportion of data frames are less than 1000 bytes in length; shorter frames result in a greater amount of overhead (back-off and frame headers) relative to actual data. The net result being lower aggregate throughput.

8.2 Concluding Remarks

Although saturation is a convenient means to model DCF performance analytically, it is an artificial system state that is surprisingly difficult to mimic in a prototype environment. Moreover, the numerical results do not explain the scalability of the protocol in typical usage scenarios, which is really what system architects and protocol designers require. In order to assess the real-world scalability of DCF one has to measure and model how 802.11 is typically used. In this research system usage was modelled by a D-BMAP with parameters drawn from traces. The resultant aggregate throughput that was measured in the D-BMAP experiments turned out to be significantly lower than saturation, emphasising how important the workload model is in assessing the scalability of DCF.

As the 802.11 standard continues to evolve, it lends itself as a platform for new technologies. Users adapt accordingly and the underlying workload model changes. Provided that trace data exists for the usage patterns, new parameters may be drawn for the D-BMAP workload model with relative ease. Performance can then be assessed via prototyping in the same manner outlined by this research.

CHAPTER 9

Future Work

Parameters for the D-BMAP and H-MMPP workload models were drawn from traces which were recorded using a RF sniffer placed in close proximity to several 802.11 stations. At the point of reception all data frames had already queued for access to the channel. Thus, the traces represent packet arrivals post queuing at the MAC layer. A superior performance modelling approach would involve recording Internet traffic as it was generated at each client and before packets queue in the network. Such a trace is non-existent, as it both challenging and time-consuming to collect.

In this research DCF performance was solely described by aggregate system throughput. Response time, jitter and error rate are examples of other metrics which describe different aspects of system performance. Although aggregate system throughput gives a good indication of the steady-state performance of the system, it cannot accurately describe the effect bursty Internet traffic has on system performance. This characteristic may be captured by measuring the average response time and jitter. The literature survey mentioned several analytic models that account for delay and jitter. A valuable extension to this research would be to compare the numerical results from one or more such models to experiment results from a hardware test bed.

For typical Internet workloads, this research shows that the benefit of the RTS/CTS access mechanism outweighs the overhead it adds at far fewer contending stations than predicted by all three of the analytic models. Unfortunately, this threshold occurs close to the upper bound of network size in the experiments. In order to accurately state the number of stations at which switching between access modes in an Internet access network becomes beneficial, one would have to rerun the experiments for sixteen or more stations. In addition, it would be prudent to explore the recently-added *pktgen* Linux kernel module as an alternative to using UDP sockets as a platform for experimentation.

References

[1] DSL Information [online]. July 2008. Available from: http://www.damnsmalllinux.org [cited 30/07/2008].

[2] EaKiu (Air Spy) - WiFi Spectrum Analyzer Software for OSX [online]. July 2008. Available from: http://www.cookwareinc.com/EaKiu/ [cited 30/07/2008].

[3] Gentoo Linux – Gentoo Linux News [online]. July 2008. Available from: http://www.gentoo.org [cited 29/07/2008].

[4] Gentoo Linux Documentation – Gentoo Handbook [online]. April 2008. Available from: http://www.gentoo.org/doc/en/handbook/index.xml [cited 29/07/2008].

[5] Gentoo Linux Projects – Embedded Gentoo [online]. July 2008. Available from: http://www.gentoo.org/proj/en/base/embedded [cited 29/07/2008].

[6] madwifi.org - trac [online]. June 2008. Available from: http://www.madwifi.org [cited 29/07/2008].

[7] MetaGeek — Makers of the Wi-Spy Spectrum Analyzer [online]. July 2008. Available from: http://www.metageek.net [cited 30/07/2008].

[8] TCPDUMP/LIBPCAP public repository [online]. July 2008. Available from: http://www.tcpdump.org [cited 30/07/2008].

[9] P. Abry and D. Veitch. Wavelet analysis of long-range dependent traffic. *IEEE Transactions on Information Theory*, 44:2–15, 1998.

[10] A. Adas. Traffic models in broadband networks. *IEEE Communications*, 35(7):82–89, 1997.

[11] I. Antoniou, V. Ivanov, V. V. Ivanov, and P. Zrelov. On the log-normal distribution of network traffic. *Physica D*, 167:72–85, 2002.

[12] P. Anvin. PXELINUX - SYSLINUX for network boot [online]. July 2008. Available from: http://syslinux.zytor.com/pxe.php [cited 29/07/2008].

[13] S. Asherson. End-to-End Security Mechanisms for the Optimised Link State Routing Protocol for Wireless Ad Hoc Networks. Msc, University of Cape Town, Private Bag X3, Rondebosch 7701, South Africa, December 2008.

[14] G. Bianchi. IEEE 802.11 saturation throughput analysis. *IEEE Communications Letters*, 2(12):318–320, 1998.

[15] G. Bianchi, A. Di Stefano, C. Giaconia, L. Scalia, G. Terrazzino, and I. Tinnirello. Experimental assessment of the backoff behavior of commercial ieee 802.11b network cards. pages 1181–1189, May 2007.

[16] G. Bianchi and I. Tinnirello. Remarks on IEEE 802.11 DCF Performance Analysis. *IEEE Communications Letters*, 9(8):765–767, 2005.

[17] J. Cao, W. S. Clevel, D. Lin, and D. X. Sun. Internet traffic tends toward Poisson and independent as the load increases. *Nonlinear Estimation and Classification*, pages 83–110, 2002.

[18] D. Cavin, Y. Sasson, and A. Schiper. On the accuracy of MANET simulators. pages 38–43, 2002.

[19] P. Chatzimisios, V. Vitsas, and A. Boucouvalas. Throughput and delay analysis of IEEE 802.11 protocol. In *Networked Appliances, IEEE 5th International Workshop on*, 2002.

[20] H. Chen and Y. Li. Performance model of IEEE 802.11 DCF with variable packet length. *Communications Letters, IEEE*, 8(3):186–188, 2004.

[21] J. Choi, J. Yoo, S. Choi, and C. Kim. Eba: An enhancement of the IEEE 802.11 DCF via distributed reservation. *Mobile Communication, IEEE Transactions*, 4(4):378–390, 2005.

[22] F. Daneshgaran, M. Laddomada, F. Mesiti, and M. Mondin. Unsaturated Throughput Analysis of IEEE802.11 in Presence of Non Ideal Transmission Channel and Capture Effects. *IEEE Transactions on Wireless Communication*, 2008.

[23] P. di Torino. MMPP based traffic generator, ns-2 code [online]. September 2008. Available from: http://www.telematica.polito.it/muscariello/mmpp_tg/ [cited 03/09/2008].

[24] P. di Torino. Tstat - TCP STatistic and Analysis Tool [online]. September 2008. Available from: http://tstat.tlc.polito.it/index.shtml [cited 03/09/2008].

[25] K. Duffy, D. Malone, and D. J. Leith. Modeling the 802.11 Distributed Coordination Function in Non-Saturated Conditions. *IEEE Communications Letters*, 9(8):715–717, 2005.

[26] M. B. Eisen, P. T. S. P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences of the United States of America*, 95(25):14863–14868, 1998.

[27] S. Floyd and V. Paxson. Why We Don't Know How To Simulate The Internet. In *Winter Simulation Conference*, pages 1037–1044, 1997.

[28] V. Frost and B. Melamed. Traffic modeling for telecommunications networks. *IEEE Communications*, 32(3):70–81, 1994.

[29] M. Garetto and C. Chiasserini. Performance Analysis of the 802.11 Distributed Coordination Function under Sporadic Traffic. *Lecture Notes in Computer Science*, 3462:1343–1347, 2005.

[30] M. Gast. *802.11 Wireless Networks: The Definitive Guide*. O'Reilly & Associates, Inc., Sebastopol, CA, USA, 2 edition, 2005.

[31] J. Hartigan and M. Wong. A k-means clustering algorithm. *Journal of the Royal Statistical Society: Series A. Statistics in Society*, 28(1):100–108, 1979.

[32] U. Herzog. Performance evaluation and formal description. In V. Monaco and R. Negrini, editors, *Advanced Computer Technology, Reliable Systems and Applications*, pages 750—756. IEEE Comp. Soc. Press, 1991.

[33] G. Holland and N. Vaidya. Analysis of TCP Performance over Mobile Ad Hoc Networks. *Wireless Networks*, 8(2/3):275–288, 2002.

[34] H. Hurst. Long-term storage capacity of reservoirs. *Transactions of the American Society of Civil Engineers*, 1952.

[35] L. Hyafil and R. L. Rivest. Constructing Optimal Binary Decision Trees is NP Complete. *Information Processing Letters*, 5(1):15–17, May 1976.

[36] D. Johnson and A. Lysko. Overview of the Meraka wireless grid test bed for evaluation of ad-hoc routing protocols. In *Southern African Telecommunications and Networks Conference*, 2007.

[37] J. T. Kaba and D. R. Raichle. Testbed on a desktop: strategies and techniques to support multi-hop MANET routing protocol development. In *MobiHoc '01: Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, pages 164–172, 2001.

[38] T. Karagiannis, M. Molle, and M. Faloutsos. Long-Range Dependence: Ten Years of Internet Traffic Modeling. *IEEE Internet Computing*, 8(5):57–64, 2004.

[39] A. Klemm, C. Lindemann, and M. Lohmann. Modelling traffic using the batch Markovian arrival process. *Performance Evaluation*, 54:149–173, 2003.

[40] D. Kotz and T. Henderson. Crawdad : A community resource for archiving wireless data at dartmouth [online]. June 2008. Available from: http://crawdad.cs.dartmouth.edu [cited 29/07/2008].

[41] W. Leland, M. Taqqu, W. Willinger, and D. Wilson. On the self-similar nature of ethernet traffic (extended version). *Networking, IEEE/ACM Transactions on*, 2(1):1–15, 1994.

[42] J. Little. A proof of the queueing formula l = λw. *Operations Research*, 9:383–387, 1961.

[43] D. Lucantoni. New results on the single server queue with a Batch Markovian Arrival Process. *Stochastic Models*, 3(1):1–46, 1991.

[44] L. Muscariello, M. Mellia, M. Meo, M. A. Marsan, and R. L. Cigno. Markov models of internet traffic and a new hierarchical MMPP model. *Computer Communications*, 28:1835–1851, 2005.

[45] Q. Ni, T. Li, T. Turletti, and Y. Xiao. Saturation throughput analysis of error-prone 802.11 wireless networks. *RRM for Next-Generation Wireless and Mobile Communication Systems*, 5(8):945–956, 2005.

[46] I. Norros. On the use of fractional brownian motion in the theory of connectionless networks. *IEEE Journal of Selected Areas in Communications*, 13(6):953–962, 1995.

[47] M. Papadopouli. UNC/FORTH Archive of Wireless Traces, Models, and Tools [online]. June 2008. Available from: http://netserver.ics.forth.gr/datatraces [cited 29/07/2008].

[48] C. G. Park, D. H. Han, and S. J. Ahn. Performance analysis of MAC layer protocols in the IEEE 802.11 wireless LAN. *Telecommunication Systems*, 33(1 to 3):233–253, 2006.

[49] V. Paxson and S. Floyd. Wide area traffic: the failure of Poisson modeling. *Networking, IEEE/ACM Transactions on*, 3(3):226–244, 1995.

[50] D. Qiao and K. Shin. UMAV: a simple enhancement to the IEEE 802.11 DCF. In *System Sciences*, page 9, 2005.

[51] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu, and M. Singh. Overview of the ORBIT radio grid testbed for evaluation of next-generation wireless network protocols. In *Wireless Communications and Networking Conference, IEEE*, volume 3, pages 1664– 1669, 2005.

[52] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu, and M. Singh. Overview of the ORBIT radio grid testbed for evaluation of next-generation wireless network protocols. In *Wireless Communications and Networking Conference, 2005 IEEE*. IEEE, 2005.

[53] J. W. Roberts. Traffic Theory and the Internet. *Communications, IEEE*, 39(1):94–99, 2001.

[54] J. Robinson and T. Randhawa. Saturation Throughput Analysis of IEEE 802.11e Enhanced Distributed Coordination Function. *Selected Areas in Communications, IEEE Journal on*, 22(5):917–928, 2004.

[55] B. Sinclair. Solving discrete-time Markov chains [online]. Available from: http://cnx.org/content/m10835/latest/ [cited 09/09/2008].

[56] A. Symington. DCF Performance Tools - Trac [online]. August 2008. Available from: http://teddy.cs.uct.ac.za:8000/dcf-perftools [cited 30/07/2008].

[57] K. Szczypiorski and J. Lubacz. Saturation Throughput Analysis of IEEE 802.11g (ERP-ORDM) Networks. *Personal Wireless Communications, IFIP Journal on*, 245:196–205, 2007.

[58] L. Torvalds. The Linux Kernel Archives [online]. 2008 July. Available from: http://www.kernel.org [cited 30/07/2008].

[59] D. Veitch. Darryl Veitch (homepage) [online]. September 2008. Available from: http://www.cubinlab.ee.unimelb.edu.au/ darryl/ [cited 03/09/2008].

[60] V. Vishnevsky and A. Lyakhov. 802.11 lans: Saturation throughput in the presence of noise. *Lecture Notes in Computer Scienceg*, 2345:1008–1019, 2002.

[61] V. Vishnevsky and A. Lyakhov. IEEE 802.11 wireless LAN: Saturation throughput analysis with seizing effect consideration. *Cluster Computing*, 5:133–144, 2002.

[62] L. O. Walters. A web browsing workload model for simulation. Master's thesis, University of Cape Town, May 2004.

[63] Y. Wang and J. Garcia-Luna-Aceves. Performance of collision avoidance protocols in single-channel ad hoc networks. In *Network Protocols, IEEE Conference on*, 2002.

[64] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson. Self-similarity through high-variability: statistical analysis of Ethernet LAN traffic at the source level. *IEEE/ACM Transactions on Networking*, 5(1):71–86, 1997.

[65] H. Wu, Y. Peng, K. Long, S. Cheng, and J. Ma. Performance of reliable transport protocol over IEEE 802.11 wireless LAN: analysis and enhancement. *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, 2:599–607, 2002.

[66] Y. Xiao. Performance analysis of IEEE 802.11e EDCF under saturation condition. *Communications, IEEE International Conference on*, 1:170–174, 2004.

[67] Q. Yu, Y. Mao, T. Wang, and F. Wu. Hurst parameter estimation and characteristics analysis of aggregate wireless LAN traffic. In *Circuits and Systems, 2005. Proceedings. 2005 International Conference on*, volume 1, pages 339–345, 2005.

[68] Y. Zheng and K. Lu. Performance Analysis of IEEE 802.11 DCF in Imperfect Channels. *IEEE Transactions on Vehicular Technology*, 55(5):1648–1656, 2006.

[69] E. Ziouva and T. Antonakopoulos. CSMA/CA performance under high traffic conditions: throughput and delay analysis. *Computer Communications*, 25:313–321, 2002.

APPENDIX A

Modulation in IEEE 802.11a, 802.11b and 802.11g

	IEEE 802.11b		IEEE 802.11g		IEEE 802.11a	
Rate	Mandatory	Optional	Mandatory	Optional	Mandatory	Optional
1	DSSS		DSSS			
2	DSSS		DSSS			
5.5	HR/DSSS	PBCC	HR/DSSS	PBCC		
6			ERP-OFDM	DSSS-OFDM	OFDM	
9				ERP-OFDM, DSSS-OFDM		OFDM
11	HR/DSSS	PBCC	HR/DSSS	PBCC		
12			ERP-OFDM	DSSS-OFDM	OFDM	
18				ERP-OFDM, DSSS-OFDM		OFDM
22				PBCC		
24			ERP-OFDM	DSSS-OFDM	OFDM	
33				PBCC		
36				ERP-OFDM, DSSS-OFDM		OFDM
48				ERP-OFDM, DSSS-OFDM		OFDM
54				ERP-OFDM, DSSS-OFDM		OFDM

Figure A.1: Modulation schemes offered by 802.11a, 802.11b and 802.11g

(Diagram adapted from http://zone.ni.com/devzone/cda/tut/p/id/7131)

APPENDIX B

802.11 Frame Transmission Time Parameters

DATA	802.11			802.3			
Component	Preamble	PLCP	MAC	LLC / SNAP	IP	UDP	DATA
Bytes	Variable		28	8	20	8	0 to 4095*
OSI Layer	Physical		Data Link		Network	Session	...

RTS	802.11		
Component	Preamble	PLCP	MAC
Bytes	Variable		20
OSI Layer	Physical		Data Link

CTS or ACK	802.11		
Component	Preamble	PLCP	MAC
Bytes	Variable		14
OSI Layer	Physical		Data Link

Figure B.1: IEEE 802.11 DATA, RTS, CTS and ACK Frames

Table B.1: OFDM data bits per symbol

Rate (Mbps)	N_{bps}	**Rate (Mbps)**	N_{bps}
6	24	24	96
9	36	36	144
12	48	48	192
18	72	54	216

APPENDIX C

Mini-ITX Client Stations

The BDS Mini-ITX system, depicted in Figure C.1, is a small and low-cost personal computing platform imported from Taipei. Its 1GHz CPU and 128 MB of RAM allow rapid booting and large RAM filesystems, which makes the development process easier. The enclosure is made of lightweight aluminium and has a removable standing brace, which allows the system to be seated normally or on its side. The whole system consumes a maximum of 1.5A at 220V, which means that significantly less current is drawn from the circuit compared to standard computers. This allows one to string a number of units together on one power lead without risk of tripping the electrical board.

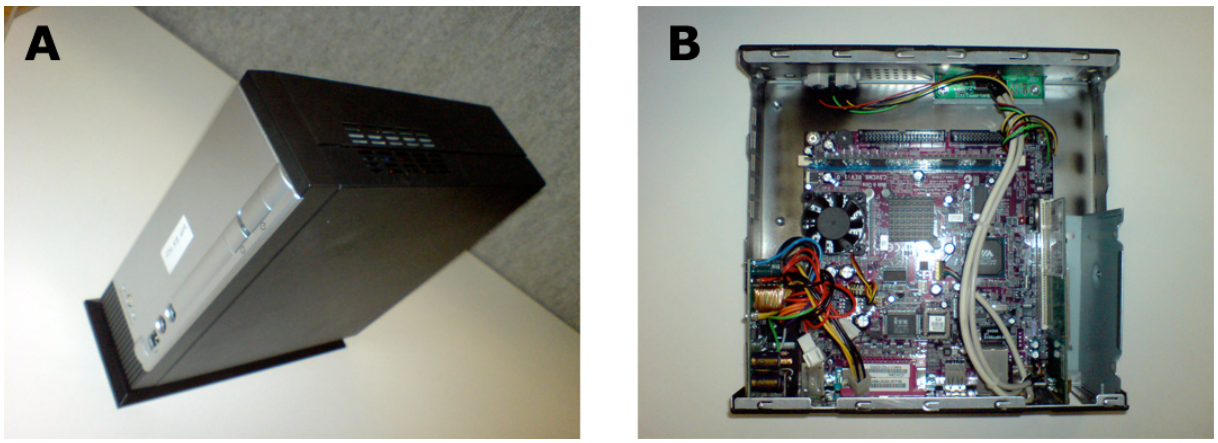

Figure C.1: External (A) and internal (B) view of the BDS Mini-ITX computer

Hardware specifications - VIA C3 Nemiah Processor, 128 MB RAM, VIA Technologies VT8633 Host Bridge with Dual IDE and single PCI v2.2 expansion, Apollo CLE266 video, USB 2.0, Realtek RTL-8139/8139C/8139C+ 100Mbps Ethernet (PXE supported), AC97 Audio.

APPENDIX D

Saturation Results

Table D.1: Analytic and experimental saturation results (normalised througput)

Basic	Stations	**Bianchi**	**Szczypiorski**	**Wu**	**Experiment**	**95% Conf.**
	1	0.439791	0.438584	0.439791	0.429769	±0.41%
	2	0.462044	0.451129	0.457838	0.467327	±0.50%
	3	0.460486	0.447090	0.452919	0.459573	±0.48%
	4	0.454803	0.440848	0.44463	0.441750	±0.31%
	5	0.448679	0.434649	0.436318	0.419651	±0.49%
	6	0.442942	0.428919	0.428616	0.397232	±0.33%
	7	0.437738	0.423698	0.421578	0.381764	±0.54%
	8	0.433037	0.418939	0.415132	0.365140	±0.72%
RTS/CTS	Stations	**Bianchi**	**Szczypiorski**[1]	**Wu**	**Experiment**	**95% Conf.**
	1	0.336896	0.335972	0.336896	0.339809	±0.46%
	2	0.356977	0.349606	0.354411	0.375137	±0.46%
	3	0.362150	0.351909	0.357363	0.377159	±0.41%
	4	0.363698	0.351773	0.357111	0.374320	±0.32%
	5	0.364013	0.350915	0.355957	0.366919	±0.27%
	6	0.363839	0.349833	0.354555	0.359284	±0.30%
	7	0.363455	0.348701	0.353116	0.351867	±0.24%
	8	0.362977	0.347583	0.351713	0.346330	±0.49%

[1] As part of this work Szczypiorski's original model was extended to support RTS/CTS

APPENDIX E

D-BMAP Results

Table E.1: D-BMAP Experiment Results (normalised throughput)

Basic	Stations	**Low**	**High**	**Avg**	**Avg (Kbps)**	**95% Conf.**
	1	0.00E+00	1.31E-04	1.93E-05	1.043442	±95.84%
	2	0.00E+00	6.07E-05	2.33E-05	1.260468	±46.04%
	3	0.00E+00	1.37E-04	3.09E-05	1.67076	±69.20%
	4	0.00E+00	1.67E-04	5.06E-05	2.73105	±52.61%
	5	1.54E-05	2.40E-04	5.69E-05	3.072438	±51.38%
	6	0.00E+00	1.89E-04	7.85E-05	4.239216	±43.07%
	7	0.00E+00	3.23E-04	8.11E-05	4.37994	±55.64%
	8	0.00E+00	2.19E-04	8.53E-05	4.605606	±36.89%
RTS/CTS	Stations	**Low**	**High**	**Avg**	**Avg (Kbps)**	**95% Conf.**
	1	0.00E+00	9.76E-05	2.03E-05	1.094958	±78.55%
	2	0.00E+00	7.34E-05	1.54E-05	0.82917	±82.85%
	3	0.00E+00	8.62E-05	2.21E-05	1.191996	±60.69%
	4	0.00E+00	1.59E-04	3.39E-05	1.828764	±64.86%
	5	0.00E+00	2.38E-04	5.48E-05	2.957796	±57.57%
	6	4.40E-06	1.61E-04	7.02E-05	3.788532	±31.83%
	7	1.35E-05	1.86E-04	8.36E-05	4.515804	±35.38%
	8	0.00E+00	2.70E-04	8.26E-05	4.462776	±59.52%

APPENDIX F

MADWifi Parameters

Table F.1: Configuring the MadWiFi interface for the experiments

Option	Shell command	Description
Antenna configuration	sysctl -w dev.wifi0.diversity=0	Turn off diversity
	sysctl -w dev.wifi0.txantenna=1	Use antenna 1 to transmit
	sysctl -w dev.wifi0.rxantenna=1	Use antenna 1 to receive
PHY/MAC configuration	iwpriv ath0 wmm 0	Turn of 802.11e QoS
	iwpriv ath0 abolt 0	Turn off Atheros extensions
	iwconfig ath0 mode ad-hoc	Set to ad-hoc mode
	iwconfig ath0 essid dnamesh	Use SSID 'dnamesh'
	iwconfig ath0 txpower 5	Set ransmit power to 5dBm
	iwconfig ath0 channel 1	Transmit on channel 1
	sysctl -w dev.wifi0.slottime=9	Set the slot time to $9\mu s$
	iwpriv ath0 mode 11g	Set mode to 802.11g
	iwpriv ath0 pureg 1	Use only OFDM rates
	iwpriv ath0 protmode 0	No CTS-to-self protection
	iwpriv ath0 shpreamble 1	Use the short preamble
Experiment configuration	iwconfig ath0 rate 54M fixed	Set the data rate to 54 Mbps
	iwpriv ath0 mcast_rate 24000	Set the basic rate to 24 Mbps
	iwconfig ath0 rts off	Turn RTS/CTS off

APPENDIX G

Workload Model Results : D-BMAP versus H-MMPP

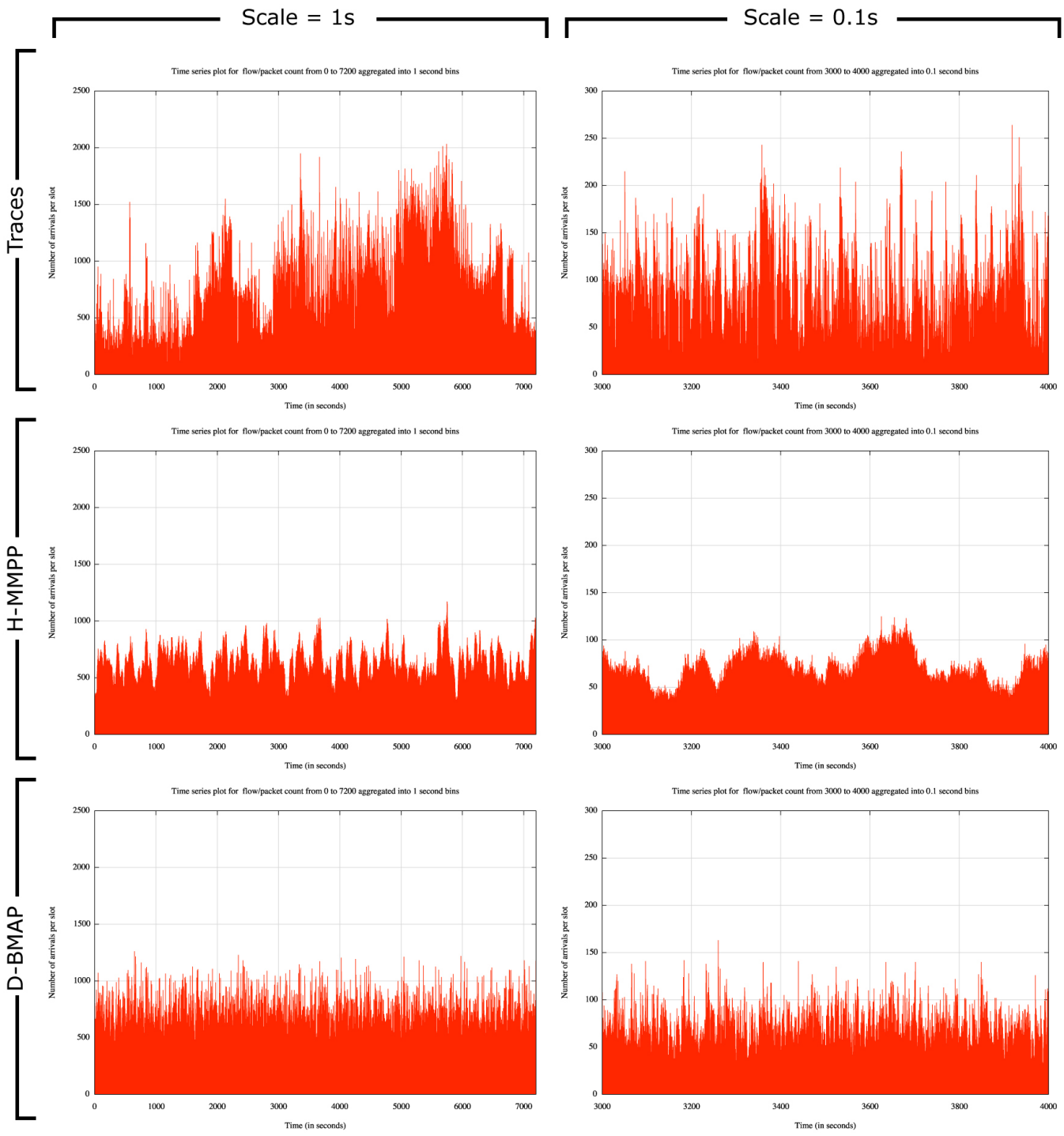

Figure G.1: Aggregate packet arrivals for 3571 clients at 1s and 0.1s scales

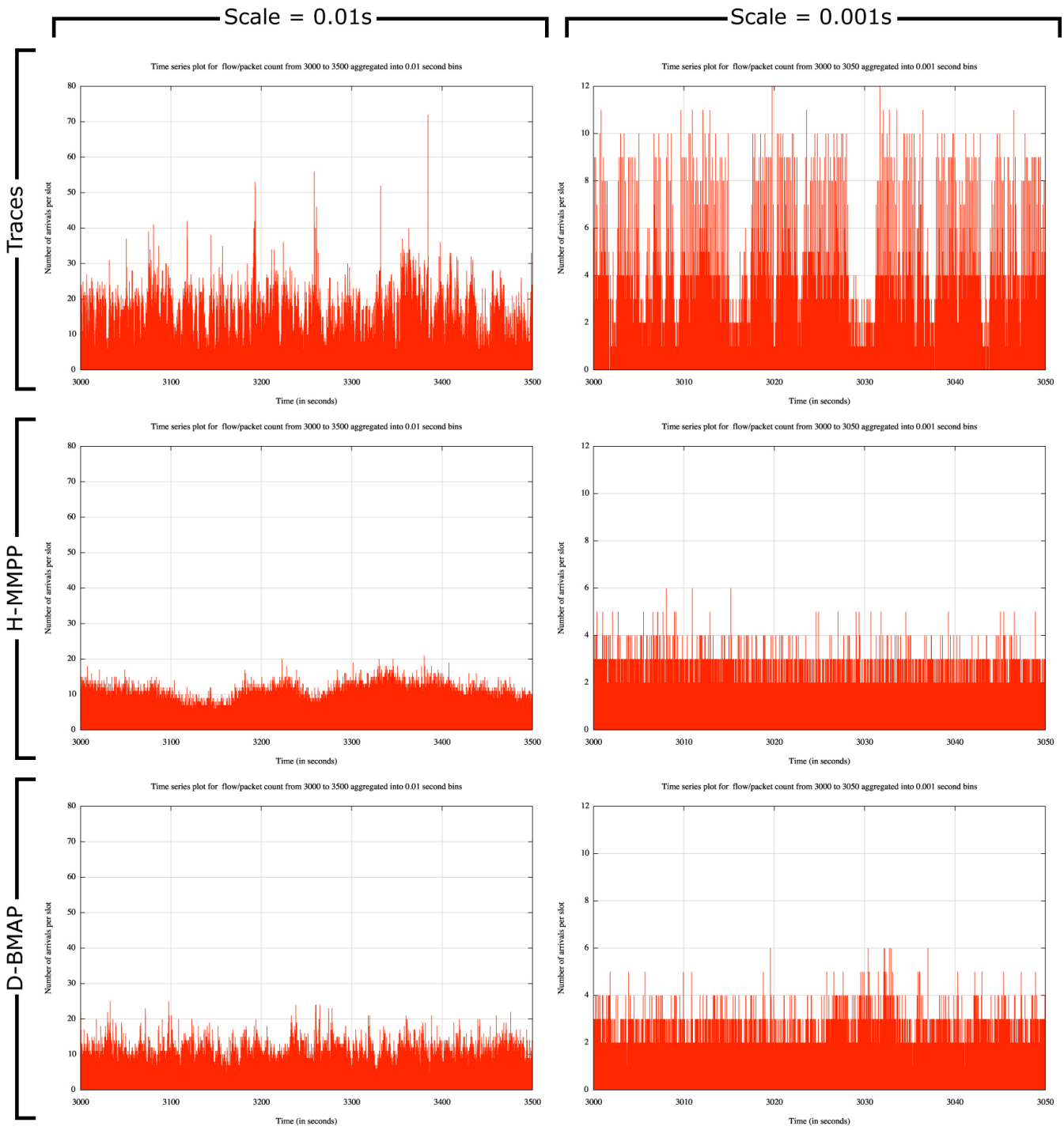

Figure G.2: Aggregate packet arrivals for 3571 clients at 0.01s and 0.001s scales

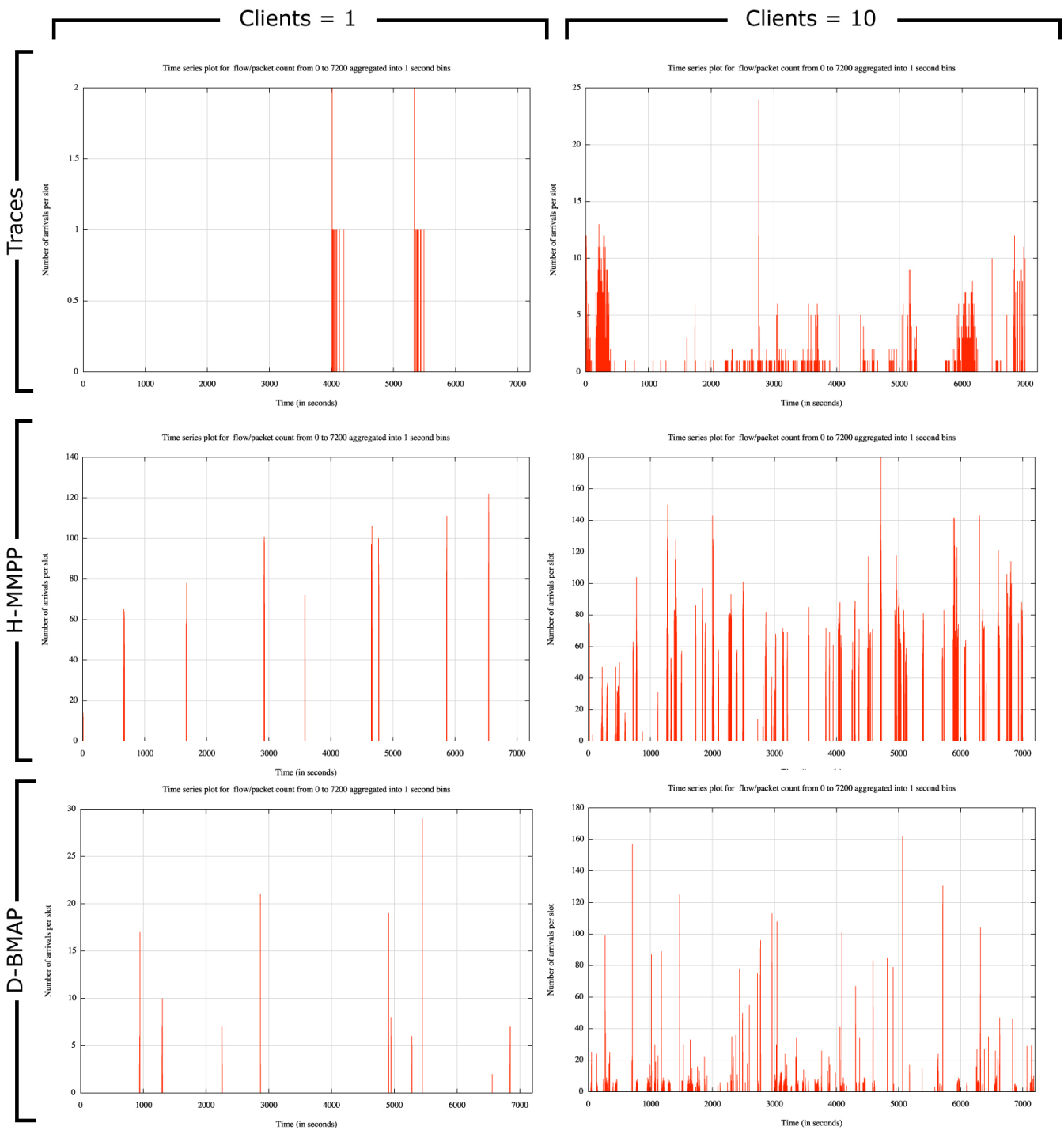

Figure G.3: Aggregate packet arrivals for 1 and 10 clients

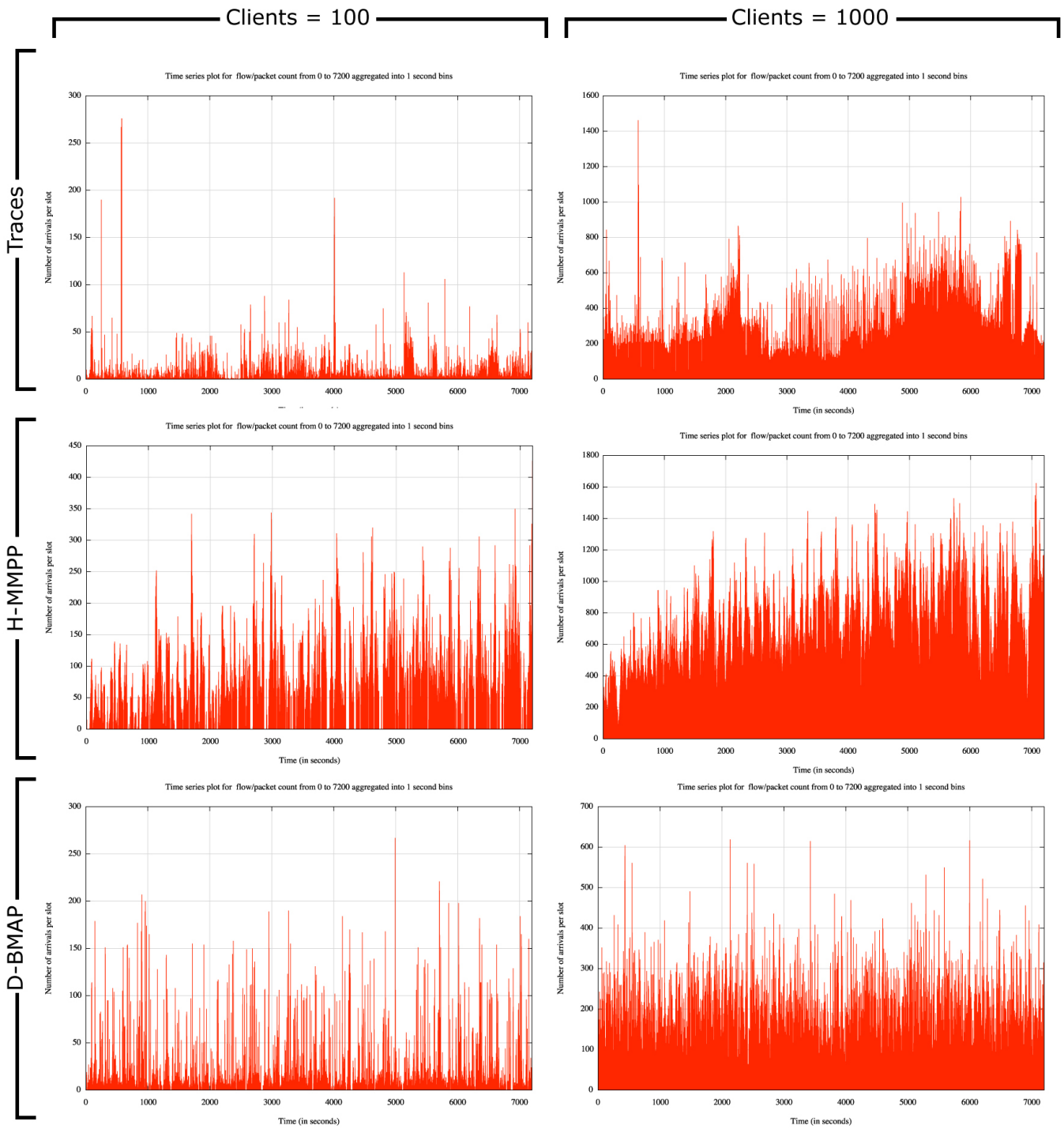

Figure G.4: Aggregate packet arrivals for 100 and 1000 clients