



**University of Crete**  
**Computer Science Department**

**Modeling, forecasting, and application-based  
characterization of traffic in campus-wide  
wireless networks**

Master thesis

**Manolis Ploumidis**

Heraklion  
February 2008

University of Crete

Computer Science Department

**MODELING, FORECASTING, AND APPLICATION BASED  
CHARACTERIZATION OF TRAFFIC IN CAMPUS-WIDE WIRELESS  
NETWORKS**

A thesis submitted in partial satisfaction of the  
requirements for the degree Master of Science in Computer  
Science

**Author:**



---

Manolis Ploumidis  
Computer Science Department, University of Crete, Hellas

**Committee members:**



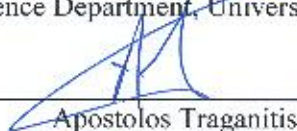
---

Maria Papadopoulou  
Assistant Professor  
Computer Science Department, University of Crete, Hellas  
Committee chair



---

Vasilios Siris,  
Assistant Professor  
Computer Science Department, University of Crete, Hellas



---

Apostolos Traganitis  
Professor  
Computer Science Department, University of Crete, Hellas

**The thesis of Manolis Ploumidis is approved:**



---

Panos Trahanias  
Professor  
Director of Graduate Studies  
Computer Science Department, University of Crete, Hellas

Heraklion, March 2008





# TABLE OF CONTENTS

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	The emergence of wireless networks	3
1.2	Motivation	4
1.3	Goals	5
1.4	Challenges	7
1.5	Related work	8
1.6	Contributions	10
1.7	Related publications	11
1.8	Roadmap	12
<b>2</b>	<b>Wireless network infrastructures</b>	<b>14</b>
2.1	UNC campus wireless infrastructure	14
2.2	Heraction Student Wireless MESH Network (HSWN)	15
<b>3</b>	<b>Data acquisition process</b>	<b>17</b>
3.1	Syslog	17
3.2	SNMP	20
3.2.1	Background on SNMP	20
3.2.2	Using SNMP in UNC wireless infrastructure	23
3.3	Packet header data	24
3.4	Datasets acquired	25

<b>4</b>	<b>Wireless traffic characterization and short-term forecasting . . .</b>	<b>28</b>
4.1	Traffic load notation . . . . .	29
4.1.1	Time series extraction . . . . .	29
4.2	Hotspot APs and their spatial locality . . . . .	31
4.2.1	Hotspot definition . . . . .	32
4.3	Basic wireless traffic characterization . . . . .	33
4.3.1	Traffic and client distribution across APs . . . . .	33
4.3.2	Correlation between total traffic and number of distinct clients per AP . . . . .	38
4.4	Wireless traffic modeling and forecasting methodology . . . . .	41
4.4.1	Spectrum analysis . . . . .	42
4.4.2	Forecasting algorithms based on wireless traffic models . . .	43
4.5	Evaluation of the performance of the forecasting algorithms . . . .	45
4.5.1	Metrics: prediction error ratio and percentage of correct predictions . . . . .	45
4.5.2	Forecasting using historical means and recent traffic (P1, P2, P3) . . . . .	46
4.6	Client level forecasting . . . . .	49
4.7	Discussion . . . . .	53
4.7.1	Other forecasting approaches . . . . .	53
<b>5</b>	<b>Multi-level application-based traffic characterization . . . . .</b>	<b>61</b>
5.1	Data preprocessing . . . . .	63
5.1.1	Reconstructing transport layer flows from packet header data	63

5.1.2	Classifying flows to application types . . . . .	64
5.1.3	Correlating packet header data with wireless sessions . . . . .	70
5.2	Aggregate traffic characterization . . . . .	72
5.2.1	Identifying dominant and most popular application types . . . . .	73
5.2.2	Distribution of flow sizes across application types . . . . .	75
5.2.3	Comparative study with other wired and wireless infra- structures . . . . .	78
5.3	Client traffic characterization . . . . .	80
5.3.1	Home application type per client . . . . .	80
5.3.2	Effect of wireless network on client application usage patterns . . . . .	81
5.3.3	Distribution of client traffic per application . . . . .	83
5.3.4	Effect of mobility on application usage patterns . . . . .	85
5.4	Application usage patterns across APs and buildings . . . . .	87
5.4.1	Application usage patterns across APs . . . . .	89
5.4.2	Home application type per AP . . . . .	90
5.4.3	Application usage patterns across buildings and building types . . . . .	92
5.4.4	Download to upload asymmetry . . . . .	93
5.5	Application based characterization of wireless sessions . . . . .	95
<b>6</b>	<b>Evaluation of parametric models for wireless traffic demand . . . . .</b>	<b>97</b>
6.1	Modeling methodology . . . . .	98
6.2	Proposed models . . . . .	100

6.2.1	Background theory on biPareto distribution and on Time-varying Poisson Process . . . . .	101
6.3	Evaluation of parametric models for wireless traffic . . . . .	102
6.3.1	Fitting sessions with a time-varying Poisson process (TVPP)	103
6.3.2	Fitting flow sizes through a Bipareto distribution . . . . .	104
6.3.3	Lognormal fit for flow inter-arrivals within a session . . . . .	106
6.3.4	Evaluating biPareto fit for number of flows per session . . . . .	107
6.4	Extending flow and session related models to the application level	110
6.4.1	Evaluating a biPareto fit for Web flow sizes . . . . .	111
6.4.2	Number of Web flows per session . . . . .	112
6.4.3	Evaluating a lognormal fit for the Web flow inter-arrival times within a session . . . . .	113
6.5	Implementation of a synthetic trace generator . . . . .	114
6.6	Discussion . . . . .	119
6.6.1	Generating synthetic traces . . . . .	119
6.6.2	Simulation testbed . . . . .	120
6.6.3	Benchmarks . . . . .	121
6.6.4	Evaluation . . . . .	122
<b>7</b>	<b>Conclusions . . . . .</b>	<b>123</b>
<b>8</b>	<b>Future work . . . . .</b>	<b>125</b>
	<b>Appendices . . . . .</b>	<b>127</b>
A	Mutliple Linear Regression . . . . .	127



B	Least Squares Method . . . . .	128
	B.1 Problem Statement . . . . .	128
	B.2 Solving the least squares problem . . . . .	129
C	Grubb's test . . . . .	131
D	Maximum Likelihood Estimate . . . . .	133
E	Source code for specific tasks . . . . .	134
	E.1 QQplots with simulation envelope . . . . .	134
	E.2 Extract basic traffic time-series from SNMP . . . . .	138
	E.3 Process time series and handle missing values . . . . .	149
	E.4 Synthetic trace generator . . . . .	154
	<b>References . . . . .</b>	<b>157</b>

## LIST OF FIGURES

2.1	UNC 802.11 wireless infrastructure . . . . .	15
2.2	HSWN wireless infrastructure . . . . .	16
3.1	Syslog collection process . . . . .	19
3.2	SNMP managed network . . . . .	21
3.3	SNMP trap . . . . .	23
3.4	Monitor point for packet header collection for wired and wireless VLANs of UNC campus . . . . .	25
4.1	Distribution of total traffic (GBytes) across APs . . . . .	34
4.2	Quantile-quantile plot for total traffic per AP . . . . .	35
4.3	CCDF of average hourly utilization per AP . . . . .	36
4.4	Number of distinct clients per AP . . . . .	37
4.5	CCDF of number of visits per AP . . . . .	38
4.6	Total AP traffic vs. number of distinct clients - linear scale . . . . .	39
4.7	Total AP traffic vs. number of distinct clients - log scale . . . . .	40
4.8	Traffic load at AP 472: (a) time series (b)Power spectrum . . . . .	42
4.9	Performance of prediction algorithms P1, P2 considering all APs . . . . .	47
4.10	Performance of prediction algorithm P3 with 25% error tolerance considering all APs . . . . .	48
4.11	Mean prediction ratios for P1, P2, and P3 with weights $(a,b,c)=(1,0,0)$ for each hotspot . . . . .	49

4.12	Median prediction ratio for P1, P2 and P3 with weights $(a,b,c)=(1,0,0)$ for each hotspot . . . . .	50
4.13	Power spectrum for clients (a) 199 (b) 373 . . . . .	51
4.14	Partial Autocorrelation Function for clients (a) 199 (b) 373 . . . . .	52
4.15	Traffic load at AP 472: (a) normal quantile plot for $X_{472}(t)$ (b) normal quantile plot for $Y(t)$ . . . . .	55
4.16	Changing patters of: (a) mean, median, and quantiles of $Y(t)$ (b) standard deviation (SD) and inter-quantile range of $Y(t)$ . . . . .	56
4.17	(a) Time series for $e(t)$ (b) Partial ACF plot for $e(t)$ . . . . .	57
4.18	Mean prediction ratio for P3 and NAMSA forecasting algorithms for each hotspot . . . . .	59
4.19	Median prediction ratio for P3 and NAMSA forecasting algorithms for each hotspot . . . . .	60
5.1	Graphlets: visual representation of transport-layer interactions for various applications (originally developed in [31]) . . . . .	68
5.2	Wireless sessions and flows . . . . .	72
5.3	Popularity of application types . . . . .	76
5.4	Distribution of flow sizes per application type . . . . .	77
5.5	CCDF of flow sizes per application type. Wired vs. wireless LAN	80
5.6	CCDF of number of flows per client for P2P and Web . . . . .	83
5.7	CCDF of percentage of total traffic per application type for each client . . . . .	84
5.8	Total traffic vs. number of distinct APs visited for (a) Web (b) P2P	86

5.9	(a) CDF of percentage of total network traffic across APs. (b) CCDF of the number of distinct clients across APs. . . . .	88
5.10	CCDF of the percentage of each APs traffic per application type . . . . .	89
5.11	Percentage of AP traffic through other applications for APs with less than 50% of their traffic through Web . . . . .	90
5.12	Combined share of traffic for P2P and Web for APs with less than 75% of their traffic through Web . . . . .	91
5.13	(a) CCDF of the assymetry index for Web, P2P, and total traffic (b) Assymetry index vs. uploaded traffic . . . . .	94
6.1	Number of sessions per hour. Empirical vs. synthesized data . . . . .	103
6.2	QQ-plot of simulated vs. empirical session intearrivals . . . . .	104
6.3	CCDF of (a) theoretical vs. empirical flow sizes (b) simulated vs. empirical flow sizes . . . . .	105
6.4	Flow interarrival times: (a) CCDF of theoretical vs. empirical (b) Logonrmlal QQ-plot . . . . .	106
6.5	CCDF of number of flows per session. Theoretical vs. fitted . . . . .	107
6.6	Time series of number of flows per hour. Synthetic vs. empirical trace . . .	109
6.7	CCDF of synthesized vs. empirical data for (a) number of flows per session (b) flow sizes . . . . .	111
6.8	Timeseries of number of flows per interval. Emperical vs. synthetic . . . . .	112
6.9	CCDF of Web flow sizes . . . . .	113
6.10	CCDF of Web flow sizes. Synthetic vs. empirical data . . . . .	114
6.11	CCDF of number of Web flows per session. Theoretical vs. emperical data .	115

6.12	CCDF of theoretical vs. actual flow inter-arrival times . . . . .	116
6.13	Average throughput per hour. Synthetic vs. empirical data . . . . .	118

## LIST OF TABLES

3.1	Summary of all traces collected from all infrastructures . . . . .	26
3.2	Number of distinct MAC addresses and APs per trace . . . . .	27
3.3	Number of LAN and WAN side IPs for HWSN and UNC infrastructures per tracing period . . . . .	27
4.1	Cross-correlation coefficients between number of distinct clients and total traffic	41
4.2	Relative prediction error ratio for P1, P2 ,P3 and recent history algorithms for client traffic forecasting . . . . .	51
5.1	Percentage of total number of flows, packets and total traffic per application type . . . . .	75
5.2	Percentage of Web and P2P traffic (bytes) across four different networks . .	78
5.3	Percentage of clients per home application . . . . .	81
5.4	Percentage of clients that transfer $x\%$ of total network's Web and P2P traffic	85
5.5	Mean percentage of a client's traffic through each application type. Most mobile vs. most stationary clients . . . . .	87
5.6	Percentage of APs with a home application. . . . .	91
5.7	APs per building category and the percentage of APs with a home application.	92
5.8	Web vs. P2P traffic share across APs of building ID 22. . . . .	93
5.9	Percentage of sessions that transfer $x\%$ of their traffic through each application . . . . .	96
6.1	Summary of models for network-wide traffic demand variables . .	100
6.2	System-wide parameters for flow, and session variables . . . . .	110

6.3	Models used to generate synthetic traces . . . . .	120
-----	--	-----

## ACKNOWLEDGMENTS

This thesis is the result of a three-year working experience as graduate student at the research group for Telecommunications & Networks at the Institute of Computer Science (ICS) of the Foundation for Research and Technology Hellas (FO.R.T.H.). These years have been very instructive, giving rise to several interesting tasks more or less related to the research. First of all I would like to thank Maria Papadopouli, my supervisor, not only for her encouragement during this work, but also for letting me proceed my own way until I needed some guidance, and for being right there with the thrusters when asked!

I would also like to thank our colleagues and researcher collaborators, F.H. Campos, H. Shen, and M. Karaliopoulos, Thomas Karagiannis for their support and collaboration. Without their support, making progress in many of the research tasks in which i was involved, would have been a lot harder.

Special thanks should also be given to my colleagues at the University of Crete and Institute of Computer at F.O.R.T.H for their continuous help and support, Elias Raftopoulos, Lito Kriara, Kostas Katertzis, Sofia Nikitaki, Antonis Makrogiannakis, Stefanos Papadakis, Vaggelis Angelakis.

I also want to thank Manolis Genetzakis, Vaggelis Boutsakis, Demetres Antoniadis for helping during during writing and correcting this thesis.

It is also important to mentioned that this work was supported by a graduate fellowship from the ICS (FO.R.T.H.), which I also truly acknowledge for providing the necessary technical equipment.

Finally i would like to thank my family for their support during my studies. For those i have forgotten, sorry, send me an e-mail..



ABSTRACT OF THE THESIS

**Modeling, forecasting, and application-based  
characterization of  
traffic in campus-wide wireless networks**

by

**Manolis Ploumidis**

Master of Science in Computer Science

University of Crete, Hellas, 2008

Professor Maria Papadopouli, Chair

Wireless local area networks are increasingly being deployed in a wide variety of areas to meet the growing demand for wireless access. When compared to their wired counterparts, wireless networks experience larger delays, lower throughput and more frequent packets losses and retransmissions. Thus, developing mechanisms, such as, load balancing, capacity planning, and admission control to improve their performance will become more and more important. In this context, it is critical to understand and analyze the performance and workload characteristics of wireless networks in order to develop wireless networks that are more scalable, robust, easier to manage and able to utilize their scarce resources more efficiently. Moreover, empirical studies can be used to guide modeling efforts for wireless demand and access patterns and provide realistic models to performance analysis studies for wireless network protocols and services.

In this work, we study a large wireless infrastructure and explore the characteristics of traffic load at APs in order to derive simple models for it. We carry

measurements based on traces acquired from the wireless infrastructure of the University of North Carolina (UNC). In the first part of the study, we perform a statistical analysis on the wireless traffic load of APs to derive models that can be used in traffic load forecasting. Based on these models, we design and evaluate several simple forecasting algorithms. The traffic characterization is extended by classifying flows into application types using a graph-based method in order to avoid the inherent limitations of a port-based classification. We perform an application-based characterization of traffic at different spatial scales and compare this methodology with others in literature. Our group has proposed models for the wireless traffic demand using real-life traces collected from the wireless infrastructure at UNC. The final part of this thesis discusses a synthetic trace generator that produces synthetic traffic based on these models. By replaying such traces, researchers could analyze the performance of various wireless networking protocols under realistic traffic conditions.

## ΠΕΡΙΛΗΨΗ ΜΕΤΑΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Μοντελοποίηση, πρόβλεψη και χαρακτηρισμός με βάση τον τύπο εφαρμογής του φόρτου κίνησης σε ένα μεγάλης κλίμακας ασύρματο δίκτυο

Συγγραφέας

Εμμανουήλ Πλουμίδης

Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης

Ελλάδα, Μάρτιος 2008

Επόπτης: Καθ. Μαρία Παπαδοπούλη

Ασύρματα δίκτυα τοπικού χαρακτήρα, εγκαθιδρύονται με γρήγορους ρυθμούς σε μια ποικιλία από διαφορετικά περιβάλλοντα με σκοπό να ικανοποιήσουν την ολοένα και αυξανόμενη ζήτηση για ασύρματη πρόσβαση. Συγκριτικά με τα ενσύρματα δίκτυα, τα ασύρματα παρουσιάζουν υψηλότερες καθυστερήσεις μετάδοσης, χαμηλότερους ρυθμούς μετάδοσης δεδομένων ενώ παρατηρούνται συχνότερα φαινόμενα όπως απώλειες ή αναμεταδώσεις πακέτων. Συνεπώς, η ανάπτυξη μηχανισμών όπως η εξισορρόπηση φόρτου κίνησης, ο σχεδιασμός χωρητικότητας και ο έλεγχος χορήγησης άδειας πρόσβασης θα αποκτούν ολοένα και μεγαλύτερη σημασία. Σε αυτό το πλαίσιο είναι κρίσιμο να κατανοήσουμε και να αναλύσουμε τόσο την απόδοση όσο και τα χαρακτηριστικά φόρτου των ασυρμάτων δικτύων με σκοπό να σχεδιαστούν ασύρματα δίκτυα που θα είναι πιο εύρωστα, κλιμακώσιμα, ευκολότερα στη διαχείριση και ικανά να αξιοποιούν τους περιορισμένους τους πόρους πιο αποδοτικά. Επιπρόσθετα, εμπειρικές μελέτες μπορούν να βρουν εφαρμογή τόσο στην καθοδήγηση μελετών μοντελοποίησης του ασύρματου φόρτου και των προτύπων πρόσβασης όσο και σε

μελέτες ανάλυσης απόδοσης για πρωτόκολλα και υπηρεσίες ασύρματων δικτύων.

Σε αυτή την εργασία, μελετάμε μια μεγάλη ασύρματη υποδομή και εξερευνούμε τα χαρακτηριστικά του φόρτου κίνησης στους κόμβους πρόσβασης με σκοπό να εξάγουμε απλοϊκά μοντέλα για αυτόν. Πραγματοποιούμε μετρήσεις βασισμένες στα πραγματικά δεδομένα που έχουν συλλεχθεί από την ασύρματη υποδομή του πανεπιστημίου της Βορείου Καρολίνας. Στο πρώτο μέρος αυτής της μελέτης, εφαρμόζουμε μια στατιστική ανάλυση του ασύρματου φόρτου κίνησης στους κόμβους πρόσβασης και εξάγουμε μοντέλα που μπορούν να χρησιμοποιηθούν για την πρόβλεψη του. Με βάση αυτά τα μοντέλα, σχεδιάζουμε και αξιολογούμε την απόδοση μερικών απλών αλγορίθμων πρόβλεψης του φόρτου κίνησης. Επεκτείνουμε το χαρακτηρισμό του ασύρματου φόρτου κατηγοριοποιώντας τις ροές δεδομένων σε τύπους εφαρμογών χρησιμοποιώντας μια γραφο-θεωρητική προσέγγιση με σκοπό να αποφευχθούν οι εγγενείς περιορισμοί μιας παρόμοιας κατηγοριοποίησης που θα βασίζεται σε αριθμούς θυρών. Χαρακτηρίζουμε τον ασύρματο φόρτο κίνησης με βάση τον τύπο της εφαρμογής σε διάφορες χωρικές κλίμακες και συγκρίνουμε τη μεθοδολογία μας με άλλες παρόμοιες μελέτες. Η ομάδα μας έχει προτείνει μοντέλα για τον ασύρματο φόρτο χρησιμοποιώντας πραγματικά δεδομένα που έχουν συλλεχθεί από την ασύρματη υποδομή του πανεπιστημίου της Βορείου Καρολίνας. Στο τελευταίο σκέλος αυτής της μελέτης, γίνεται περιγραφή μίας γεννήτριας συνθετικών δεδομένων η οποία παράγει συνθετικά δεδομένα με βάση αυτά τα μοντέλα. Αναπαράγοντας τέτοιου τύπου δεδομένα, οι ερευνητές μπορούν να αναλύσουν την απόδοση διάφορων πρωτοκόλλων ασυρμάτων δικτύων κάτω από ρεαλιστικές συνθήκες.

# CHAPTER 1

## Introduction

### 1.1 The emergence of wireless networks

Wireless networks are increasingly being deployed to meet the growing demand for wireless access. IEEE 802.11 networks are becoming widely available in universities, corporations, hospitals, residential areas and generally in every type of public area. More and more users resort to wireless technologies for their every-day activities. Popular applications and services from the wired networks shift in the wireless arena while new applications are increasingly being deployed. Apart from new applications that shift in wireless networks, usage patterns of current ones evolve rapidly over time. The proportion of wireless streaming audio and video traffic increased by 405% between 2001 and 2003/2004, P2P from 5.2% in 2001 to 19.3% in 2003/4, filesystems from 5.3% to 21.5%, and streaming from 0.9% to 4.6% between January 2006 and March 2006 [52]. The increasing popularity of wireless networks has also attracted the interest of the research community. New standards for wireless technologies are being developed to meet the diverse needs for wireless access and quality of service requirements while current ones are continuously being improved. More and more researchers are becoming interested in understanding wireless network characteristics and analyzing their performance.

## 1.2 Motivation

Despite of their broad deployment and usage, the evolution of WLANs towards large multi-service networks has introduced several challenges that have not been thoroughly addresses by the research community. Wireless networks have more vulnerabilities and face more resource limitations than their wired counterparts, such as, limited throughput and larger delays. Apart from that, due to the nature of wireless channel and its dynamics, wireless clients experience more frequently phenomena, such as, disconnections from the infrastructure and packet losses. Other phenomena that degrade the performance of wireless networks are congestion due to the nature of the shared medium and interference caused by other in-proximity wireless transceivers. It is also frequent for users of a large wireless infrastructure to fall into areas with poor or no coverage known as “dead-spots”. All aforementioned limitations of wireless networks have motivated standardization efforts in an attempt to improve the management of limited radio resources and assure QoS for specific applications. Within this context, there is a growing need for mechanisms, such as, load balancing, capacity planning, admission control and resource reservation that are expected to improve significantly the performance of current WLANs. Understanding the underlying characteristics of wireless demand and characterizing traffic load and access patters is elementary to both the design and the implementation of such mechanisms. Measurement studies though, can be particularly beneficial towards this direction. It is critical to understand the performance and workload of the wireless networks and develop wireless networks that are more robust, easier to manage and scale, and able to utilize their scarce resources more efficiently.

### 1.3 Goals

The research goals of this work are twofold. In the first stage we intend to characterize and analyze traffic of an infrastructure-wide WLAN at two different network layers namely the MAC and transport layer. By analyzing traffic at the MAC layer we intend to capture real-life phenomena, such as, user communication and association patterns (e.g. periodicities) that will assist in modeling traffic and implementing traffic forecasting algorithms. Such an analysis will also provide useful information to network administrators by indicating overloaded APs or possible dead-spots. Analyzing and characterizing traffic at the MAC layer will provide us with a deeper insight into the performance of WLAN's and will guide decisions about mechanisms, such as, load balancing or admission control that should be incorporated.

Analyzing traffic at the transport layer aims at exploring how well-known transport layer protocols, such as UDP and TCP, perform over the wireless channel. This study also provides a multi-level application based characterization of transport layer traffic from the perspective of network, client, AP and building. Traffic is classified into application types using a novel tool called BLINC avoiding limitations and restrictions of a payload- or port-based classification. Such a characterization of wireless traffic is important since different application types have diverse requirements in terms of network resources and thus have a different effect on overall network performance. Our analysis can provide network administrators with useful information, such as, APs or buildings that are overload by P2P traffic, security vulnerabilities of the network, or demand for delay sensitive applications (VoIP).

In the second stage we shall focus on implementing and evaluating models for wireless traffic on two different network layers namely MAC and TCP. Such mod-

els will be useful for network administrators who wish to perform load balancing, resource reservation and resource provisioning. In order to model traffic at a finer level namely, AP and client, we exploit traffic characteristics ,such as, periodicities and access patterns observed and make use of the diverse datasets collected from our infrastructure. We use these models to implement simple short-term forecasting algorithms that can be incorporated in access points to allow them to perform admission control and capacity planning in a real-time fashion. So, in this context, each AP predicts its traffic load for the next time interval (e.g., next hour or five minute interval) and uses its traffic load estimates during admission control to not only better manage its traffic demand but also advise clients to associate with the appropriate APs to better utilize their local resources. Such predictions can be used to reduce the energy spending at the client side, improve the capacity utilization of wireless LANs, and balance traffic load across APs.

The major part of our modeling effort focuses on extracting and evaluating models for wireless traffic at the transport layer. We evaluate models that were suggested in [11] by generating synthetic traces and comparing them against original traces through statistical tests and simulations. Our intention is to embed all proposed parametric models in a software module that will generate realistic synthetic traces in a scalable manner. We shall also show that our models can be used to describe accurately Web traffic apart from aggregate network traffic. Scalable and realistic models for wireless demand can be beneficial to network administrators who wish to perform capacity planning or resource provisioning. They can also be used to guide decisions for QoS support at specific locations of the network. Finally, such models can be used in the absence of real-time testbeds to generate realistic traces and assist in performance analysis studies of wireless networks.



## 1.4 Challenges

Wireless measurements are in general more complex than those in wired networks. In wireless measurements studies, researchers are interested in a rather wide range of phenomena, such as, disconnections, mobility, energy consumption, received signal strength, interference, and offered load as opposed to studies concerning wired networks where the main focus is on actual traffic load. Wireless measurements thus, often require traces from different network layers, such as, MAC, TCP-IP or even physical layer. Apart from that, depending on how detailed view of the network is required at the spatial dimension, e.g., at an AP, APs co-located in a building or set of buildings, and the architecture of the network (single link-level subnet or multiple subnets), one may need to capture traffic at various physical locations requiring thus multiple monitor points. Many of the IEEE 802.11 MAC-layer frame sniffers that are available either lack sufficient documentation or they need a significant amount of tuning so researchers often have to build custom equipment or resort to expensive commercial tools to capture over-the-air traffic with the required level of detail. Moreover, depending on the phenomena studied, it is often required to cross-correlate diverse traces collected from the same wireless infrastructure, such as TCP, SNMP or Syslog messages. Such a correlation process may not always be straightforward requiring a lot of processing time and using complicated algorithms. It comes as no surprise that only recently traces from large-scale wireless infrastructures have been made available. Notably, the majority of the measurement studies e.g., [56, 62, 72], make high-level observations about network dynamics in both the temporal and spatial domains without getting into the detail that modeling tasks require.

## 1.5 Related work

The increased interest in wireless networks has attracted research community's interest and has led to several studies exploring a large variety of wireless network characteristics. There are several empirical-based measurement studies about traffic load that explore characteristics and patterns of wireless demand at various spatial scales [75, 61, 70, 12, 68, 49, 52]. There is also an increased interest in user access patterns and mobility trying to evaluate and construct realistic mobility models [70, 80, 28, 34, 13, 35, 36, 63]. Also, being aware of the resource limitations and wireless channel dynamics, wireless network researchers have focused on exploring the performance IEEE 802.11 MAC layer. Some of the phenomena studied are:

- Hand-off delays and disconnections from the infrastructure [37, 65];
- Packet delays and packet losses observed at the MAC layer [65];
- Signal strength quality of wireless links and routing [66, 38, 57];

IEEE802.11-based mesh networks have also received a lot of attention from researches who explore architecture and routing issues in such networks [33, 14, 15, 100, 57, 58]. Recently there have been various studies analyzing and modeling time-series traffic load in order to support forecasting algorithms or admission control and resource provisioning mechanisms [88, 81, 77, 39, 12].

Apart from characterizing or modeling wireless demand at the MAC layer, many studies have focused on characterizing wireless demand at the transport layer. One direction of such a characterization is identifying the application type for each flow. However, standard port based classification techniques used so far were proved to be inaccurate [32, 55]. Several system-oriented [31] or statistical

methods [30, 54] have been developed to address the inefficiency of current flow classification techniques. Although classifying traffic into applications has received significant attention in wired networks [3, 53, 60, 30, 54], there are only a few similar studies in the case of wireless ones [70, 52]. Tutschku examined the difference of the uploading from the downloading traffic of a popular P2P application in a wired network and reported a significant amount of uploaded P2P traffic in [59]. Guo *et al.* in [41] studied the uploaded to downloaded traffic asymmetry for bit torrent clients and concluded that it is highly affected by high speed downloading. Chambers *et al.* in [40] characterized the online games usage in terms of user sessions and periodicities of the workload. Moreover, Guha *et al.* in [17] studied the structure of the network that a famous VoIP system builds in order to distinguish it from P2P systems.

Most traffic modelling efforts focus on wired networks. Flow-level traffic variables have been the subject of modelling in various studies, embracing almost all Internet protocols and applications, such as, TCP [76, 73, 78, 86] and multimedia streaming traffic [79]. The concept of session as a structure of the user activity was used in [87] for Ftp traffic, as a synonymous of the Ftp control connection. The term was used more explicitly later in Web traffic modeling. Both empirical [74, 84] and statistical [83, 71, 49] modeling approaches have been used for the description of traffic at the two levels. A common feature of these studies is that the flow/session borders are heuristically defined by intervals of user inactivity. Although there have been several modelling efforts in the wireless domain, mainly performance analysis studies on wireless network protocols and mechanisms, they employ traffic scenarios that simulate saturation conditions [42, 7, 43, 6, 18, 19, 8, 25]. Other studies employ UDP flows of fixed packet rate among a few source and destination pairs to generate synthetic traffic [47, 10, 48, 20, 21]. There were only few studies that employed stochastic packet

rate models, such as, Uniform, Poisson, Pareto, AR, and Markov [22, 23, 24]. Example of studies using TCP flows are [46, 45, 9] while studies replaying real-life traces are [44].

## 1.6 Contributions

In this study we characterize traffic from a campus-wide WLAN at the MAC and transport layer using a rich set of datasets. Characterizing traffic at the MAC layer reveals interesting phenomena of wireless demand, such as, the distribution of traffic across APs, correlation between traffic load and number of distinct clients seen by an AP, as well as, periodic patterns in the timeseries of traffic of both clients and APs. Based on these periodic patterns we derive simple models for wireless demand that are used to build short term forecasting algorithms. Hourly predictions of the traffic load at an AP exhibit very large prediction errors due to the high variability of traffic. When moving to a finer spatial scale (client level) periodic based forecasting algorithms perform better.

We also characterize traffic at the transport layer. At first we use a novel method to classify flows into application types avoiding the known port limitation. Application usage patterns are identified at four different levels, namely client-, AP,building-, and network-level. This characterization indicates that the dominant and most popular application types across the network are Web and P2P. Also, most wireless clients appear to exhibit strong application preferences transferring the majority of their traffic through a specific application type. When compared to other wired and wireless networks, we observe a larger percentage of Web traffic, while the corresponding percentage of P2P is lower than all wired counterparts. We notice that mobility, when expressed through the number of distinct APs that a client has visited, does not affect application usage patterns

across clients.

Extending the transport layer characterization of wireless demand, we evaluate the performance of parametric models for session- and flow-level variables. Specifically, the biPareto distribution provides a very efficient fit for both flow sizes and number of flows per session. Also, the lognormal distribution consists a satisfactory approximate for flow inter-arrival times within sessions. Finally, we show the a TVPP is a suitable model for session arrivals. It is shown that these models can also be used for the case of Web flows. Based on these models, we implemented a synthetic trace generator which we prove that captures the trends observed in the real trace acquired from UNC’s WLAN.

## 1.7 Related publications

The modelling methodology presented in section 4 along with the implementation and evaluation of forecasting algorithms for wireless traffic of APs at an hourly timescale were the main contributions of the paper: **“Short-term traffic forecasting in a campus-wide wireless network”** published in *16th Annual IEEE International Symposium on Personal Indoor and Mobile Radio Communications* by Maria Papadopouli, Haipeng Shen, Elias Raftopoulos, Manolis Ploumidis, and Felix Hernandez-Campos.

The conclusions of our application-based characterization of wireless traffic at the client-, AP-, building-, and network-level that are presented in section 5 are based on the findings of the paper: **Multi-level application-based traffic characterization in a large-scale wireless network** published in *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)* by Manolis Ploumidis, Maria Papadopouli, and Thomas Karagian-

nis.

Issues concerning modelling wireless traffic at the transport layer presented in Section 6 are based on two recently published papers. The methodology used to derive parametric models for flow- and session-level variables is presented in the paper: **Spatio-Temporal Modeling of Traffic Workload in a Campus WLAN** published in *Second annual international Wireless Internet Conference* by Felix Campos, Merkourios Karaliopoulos, Maria Papadopouli, and Haipeng Shen. The evaluation of these parametric models through both simple statistical tests and simulations addressed in the same section were the main contributions of the paper: **Synthetic traffic generation based on Measurement-driven modelling of large Wireless Local Area Networks** accepted in the student demo competition of *ACM MobiCom/MobiHoc 2007* by Manolis Ploumidis, Elias Raftopoulos and Maria Papadopouli.

## 1.8 Roadmap

Section 2 provides a brief description of the two infrastructures over which traces were collected and analyzed in terms of the architecture of the network, protocols used and their sizes. Section 3 first provides some background information about the data collection mechanisms that are used. Then it describes how these mechanisms were used to acquire datasets from the two infrastructures. Finally a list of the datasets acquired is provided. Section 4 performs a characterization of wireless demand at the MAC layer. This characterization includes identifying some general workload characteristics, such as, traffic distribution across clients. It also describes how statistical properties of traffic timeseries are exploited to create simple models for wireless demand and how these models are incorporated to implement forecasting algorithms. At the final part of this section, forecasting

algorithms are evaluated at the AP and client level. Section 5 consists the first part of the characterization of wireless demand at the transport layer. It first describes the data pre-processing part which includes classification of flows into application types using BLINC. The main part of this section presents the results of a multi-level application-based characterization of UNC's wireless traffic from the perspective of the network, building, AP, and client. Finally, section 6 first provides a brief description of the modeling approach followed in [11] along with the models proposed for wireless demand. The main part of this section focuses on evaluating the performance of these models. It finally presents the basic implementation characteristics of a custom synthetic trace generator based on the models evaluated.

## CHAPTER 2

### Wireless network infrastructures

In this section we present the architecture and key characteristics of the two wireless infrastructures that were monitored. The main part of this study focuses on the 802.11 campus-wide wireless infrastructure of the University of North Carolina while the second infrastructure monitored is the 802.11 wireless student MESH network of Heraklion.

#### 2.1 UNC campus wireless infrastructure

The University of North Carolina at Chapel hill started deploying an 802.11 wireless network in 1999. Wireless access is available for almost all of the building of the 729-acre university campus including lecture halls, residences and administrative buildings. By the time measurements were made, about 500 APs provided wireless access to 26,000 students, 3,000 faculty members and 9,000 staff members and a couple of off-campus administrative offices.

The majority of the APs belong to the Cisco 1200 Aironet series while there is still a significant number of 350 series and fewer 340 series APs. Two are the main trends with respect to the infrastructure evolution with time: it is constantly growing, with APs exceeding 750 by June 2006 and, in parallel, older 340/250 series are being replaced by 1230/1240 series AP compliant with 802.11a,g [11]. Figure 2.1 represent an abstraction of UNC's wireless infrastructure. Wireless



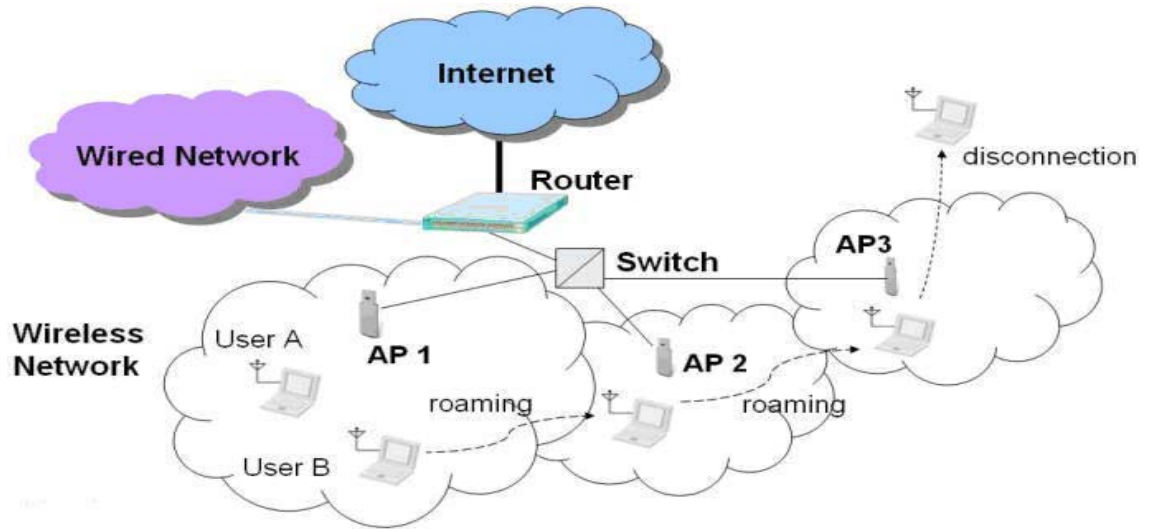


Figure 2.1: UNC 802.11 wireless infrastructure

clients roam seamlessly among APs belonging to different Basics Services Sets (BSSs) and exchange traffic with other hosts of the same WLAN or hosts residing in the Internet. Client B for example, first associates to AP 1, then roams to APs 2 and 3 and finally terminates his interaction with the infrastructure by disassociating from AP 3. UNC's WLAN interconnects to the wired component of the infrastructure through UNC's backbone network and to the rest of the Internet through campus' egress router.

## 2.2 Heraclion Student Wireless MESH Network (HSWN)

HSWN is a wireless MESH network that was founded in 2002 in Heraclion of Greece. It covers an area of 10 square kilometers providing wireless access to more than 180 members. To the best of our knowledge this is the first study that collects a wide variety of traces, such as, packet headers, SNMP, and Syslog from a wireless multi-hop MESH network. Figure 2.2 provides a brief abstrac-

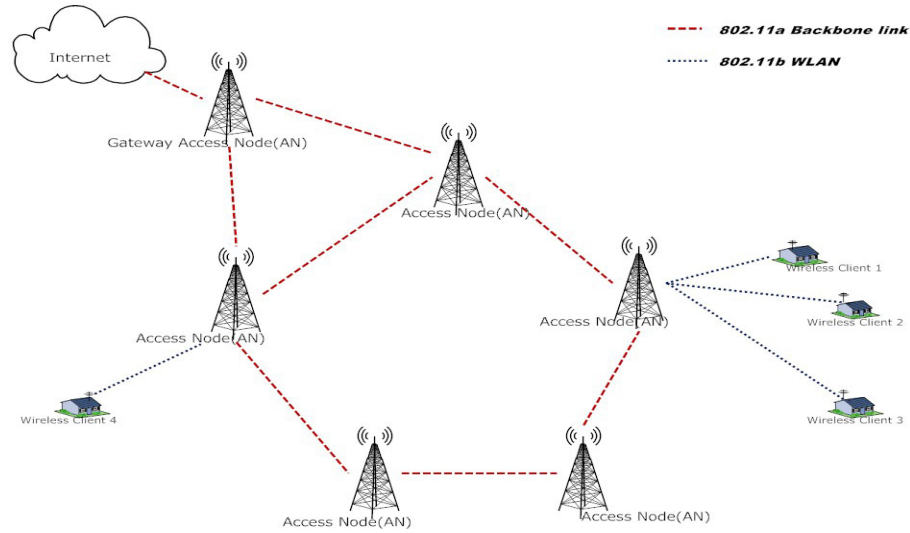


Figure 2.2: HSWN wireless infrastructure

tion of HSWN architecture. The infrastructure of HSWN consists of 10 access nodes (AN) through which users of the HSWN exchange traffic with each other and access the Internet. Each AN consists of an 802.11b AP and an 802.11a wireless interface. APs are used by clients to connect to the infrastructure and can be Cisco Aironet series 340 or D-link DWL-900+. Each AN's 802.11a interface serves as a backbone link to other ANs forming thus a multi-hop wireless MESH network. Another key difference between UNC's wireless infrastructure and HSWN is that HSWN users are stationary and connect to network's infrastructure through a directional antenna.

## CHAPTER 3

### Data acquisition process

In this section we present the monitoring mechanisms that were used to collect traces from the two wireless infrastructures studied along with some general workload characteristics. Three types of data have been used to track the interaction of clients with the wireless network infrastructure: Syslog messages, Simple Network Management Protocol (SNMP) data, and packet header traces. Each monitoring mechanism captures different aspects of wireless demand providing thus a different view of the infrastructure being monitored. Syslog messages capture the interaction of a wireless client with the infrastructure and record events such as associations or disassociations of the client from an AP while SNMP and packet header traces provide information about wireless traffic at the MAC and transport layer respectively. The first two types of data have been collected almost continuously from the wireless network, whereas the packet header traces come from two different eight-day monitoring periods spaced one year apart, i.e., April 13-20 2005 and Apr 28-May 5 2006.

#### 3.1 Syslog

The Syslog protocol, defined in RFC 3164, is a protocol that provides a transport to allow a device to send event notification messages across IP networks to event message collectors, also known as Syslog servers. The protocol is simply designed

to transport these event messages from the generating device to the collector. The collector does not send back an acknowledgment upon the reception of Syslog messages.

As various kernel modules of a UNIX based operating system generate messages that require user's or administrator's attention, on the same way devices such as routers, switches, firewalls, VPN servers, access points and other network devices, generate messages/notifications that can be used either for troubleshooting or to inform about an event that requires special attention. A hardware firewall for example may send a Syslog message to a preset Syslog server whenever it blocks a connection. All events logged by Syslog can then be used by system administrators to extract valuable information. As Figure 3.1 shows, all devices that are Syslog-compatible can be configured to send Syslog messages to a central point. Messages between the device and Syslog server are delivered through UDP so there is the case of missing some events due to loss of the notification.

The Syslog protocol also defines various levels of severity for the messages generated. Each Syslog messages arriving at the Syslog server also contains a single-digit integer indicating the severity of the message. Usually system administrators adjust the severity level during the setting up of a Syslog collection process to define the classes of messages to which they are interested in. In the case where Syslog messages are used to monitor an 802.11 infrastructure, they can be triggered by different types of events at the IEEE 802.11x MAC layer, including the (re)association/disassociation of a client with/from an AP, its authentication/deauthentication with/from the network, and a reset of a client's connection. There are seven types of events that trigger an AP to transmit a Syslog message. These messages and their corresponding events are interpreted as follows:

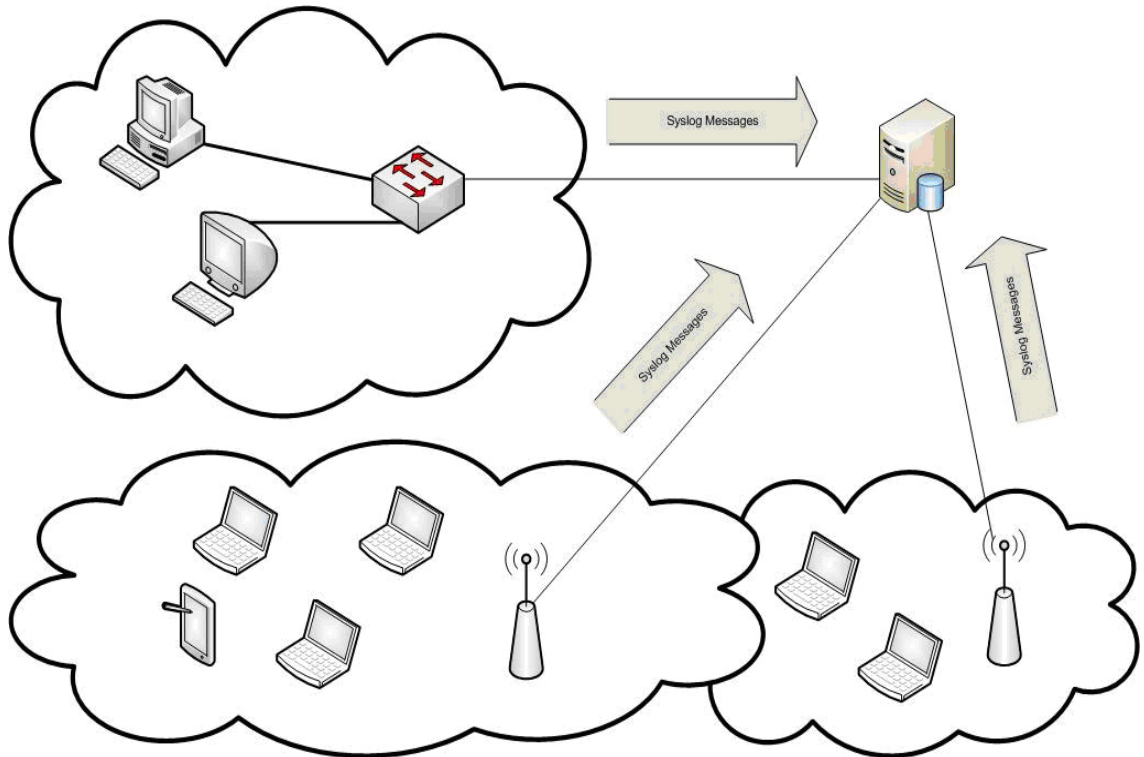


Figure 3.1: Syslog collection process

*Authenticated:* A card must authenticate itself before using the network. Since a card still has to associate with an AP before sending and receiving data, we ignore any authenticated messages.

*Associated:* After it authenticates itself, a card associates with an AP. Any data transmitted to and from the network is transmitted by the AP.

*Reassociated:* A card may reassociate itself with a new AP (usually due to higher signal strength) or the current AP. After a reassociation with an AP, any data transmitted to and from the network is transmitted by that AP.

*Roamed:* After a reassociation occurs, the old AP and sometimes the AP with which the card has just reassociated sends a roamed message. Since we still

receive the reassociated message, we can ignore this message as well.

*Reset:* When a card's connection is reset, a reset message is sent. In our trace, cards with a reset message are only involved in reset messages. We believe this to be an artifact of not having logs from all of the APs, and therefore we ignore any reset messages.

*Dissasociated:* When a card wishes to disconnect from an AP, it disassociates itself. We ignore any disassociated messages for a card if the previous message for that card was a disassociated or deauthenticated message.

*Deauthenticated:* When a card is no longer part of the network a deauthenticated message is sent. It is not unusual to see repeated deauthenticated messages for the same card, with no other type of events for that card in between. We ignore any deauthenticated messages for a card if the previous message for that card was a disassociated or a deauthenticated message. A disconnection message describes either a disassociated or deauthenticated message.

All network APs for both infrastructures were configured to report Syslog events to a server, which was continuously operational.

## 3.2 SNMP

### 3.2.1 Background on SNMP

SNMP is an Internet-standard protocol for managing devices on IP networks. Many kinds of devices support SNMP, including routers, switches, servers, workstations, printers, modem racks, and uninterruptible power supplies (UPSs). It is an application layer protocol that facilitates the exchange of management information between network devices and is part of the Transmission Control Protocol/Internet Protocol (TCP/IP) protocol suite. SNMP enables network ad-

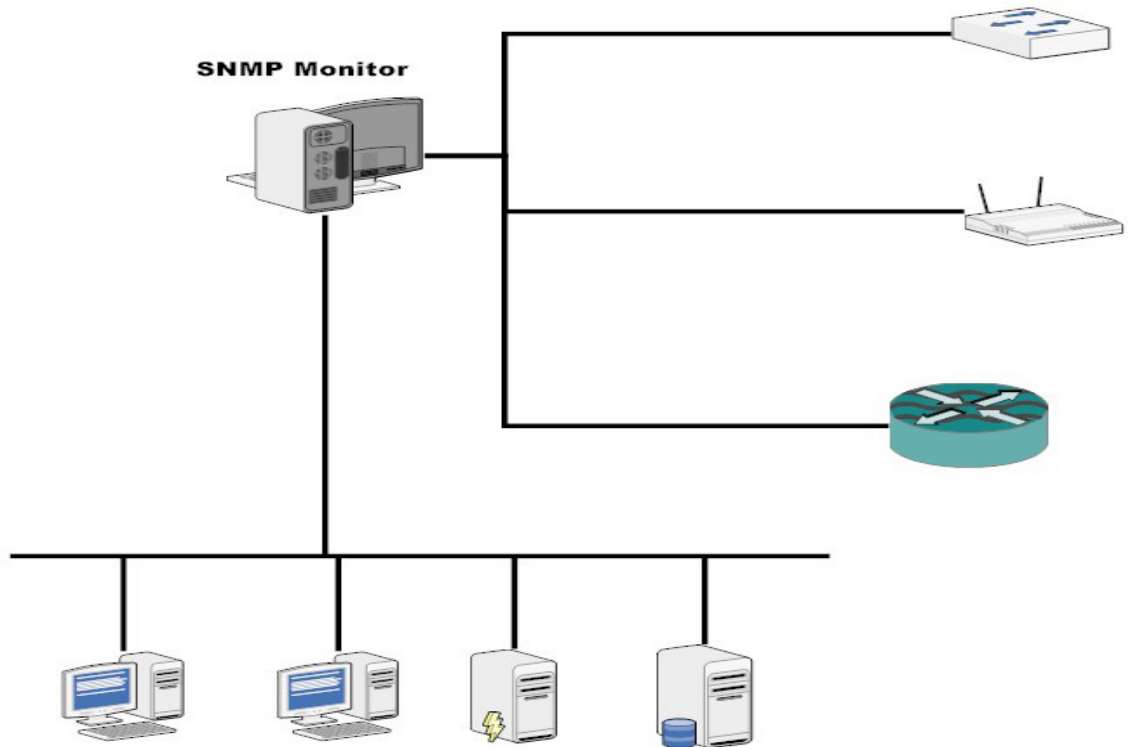


Figure 3.2: SNMP managed network

ministrators to manage network performance, find and solve network problems, and plan for network growth. Current standardization efforts have led to the release of two versions of SNMP namely SNMPv1 and SNMPv2 which describe SNMP basic components, basic commands, data representation, and manageable objects. Figure 3.2 illustrates a demonstrative network managed by the SNMP protocol.

The basic components of an SNMP-managed network are *managers* and *agents*. A manager is a server running some kind of software that can handle management tasks for a network. Managers are often referred to as *Network Management Stations (NMSs)*. A NMS is responsible for polling and receiving traps from agents in the network. A *poll*, in the context of network management,

is the act of querying an agent (router, switch, Unix server, etc.) for some piece of information. A *trap* is a way for the agent to tell the NMS that something has happened. Traps are sent asynchronously, not in response to queries from the NMS. The NMS is further responsible for performing an action based on the information it receives from the agent. Consider for example the case where there is a hardware failure at a key component of a network such as a router or a switch. The malfunctioning device can send a trap to the NMS which in turn will be responsible for taking some action like informing network administrator. The second entity, the agent, is a piece of software that runs on the network devices being monitored. It can be a separate program (a daemon, in Unix language), or it can be incorporated into the operating system (for example, Cisco's IOS on a router, or the low-level operating system that controls a UPS). Today, most IP devices come with some kind of SNMP agent built in allowing for administrators to poll the status of the device. The agent provides management information to the NMS by keeping track of various operational aspects of the device. Figure 3.3 represent the way in which an NMS and an agent interact. SNMP uses the User Datagram Protocol (UDP) as the transport protocol for passing data between managers and agents. UDP however is unreliable since it lacks an acknowledge mechanism so special care is needed for the case of missed polls.

All information that an SNMP agent can provide to the NMS is organized hierarchically in a Manageable information base (MIB). MIBs consist of manageable objects that can be polled through SNMP and identified through unique object IDs. Such manageable objects can be the total number of packets observed at a router port, total received traffic by an APs wireless interface, number of retransmitted MSDUs e.t.c.



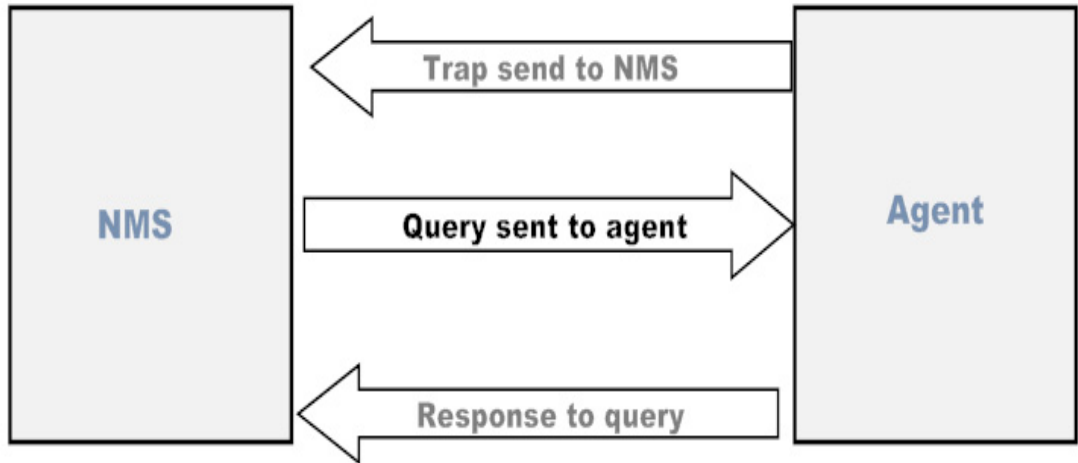


Figure 3.3: SNMP trap

### 3.2.2 Using SNMP in UNC wireless infrastructure

We implemented a custom SNMP-polling system relying on a non-blocking SNMP library to poll all APs of the two infrastructures studied. APs are polled independently with a period of five minutes, so that delays incurring during the processing of SNMP polls by the slower APs do not affect the other APs. This eliminates any extra delays due to the slow processing of SNMP polls by some of the slower APs. The system ran in a multiprocessor system and the CPU utilization in each of the three processors we employed never exceeded 70%. Note that SNMP data can be collected in two forms namely AP and client SNMP although they are derived from the same MIBs. The only difference is that client SNMP polls also provide some client related objects, such as, client up-time or client traffic per AP instead of aggregate AP traffic. Each SNMP polling returns a set of values at each time “t”, for each AP, that correspond to the variables/characteristics being monitored, such as:

1. total time that an AP is up since last reboot;
2. total number of received/sent bytes by the AP;
3. total number of received/sent unicast packets;
4. total number of successfully/undelivered MSDUs;
5. transmission rate of APs wireless interface;
6. client up time since last client's association;

### 3.3 Packet header data

While SNMP and Syslog protocols are mostly used to capture MAC layer information from a wireless infrastructure, packet header data collected with tools like *tcpdump* can provide transport layer information about the network being monitored. *Tcpdump* is a command-line tool for monitoring network traffic at the TCP/IP layers. It can capture and log all packets “sniffed” from a particular network interface or on all interfaces of the IP device being monitored [90]. *Tcpdump* like other packet sniffers contains a set of flags that allow network administrators to monitor specific IP addresses or sets of IP addresses and also to keep or discard payloads from packets. It can also store packets captured in various formats, such as, binary or Ascii. Note though, that the output of *tcpdump* is packet headers and not TCP flows or TCP connections. Reassembling of packets into flows should be accomplished in a subsequent processing step. Before using *tcpdump* however, the only parameter that requires consideration is the size of the network being monitored. Storing every packet noticed on a link is both processor and storage intensive. Especially when high-speed links are monitored, *tcpdump* may start experiencing packet losses.

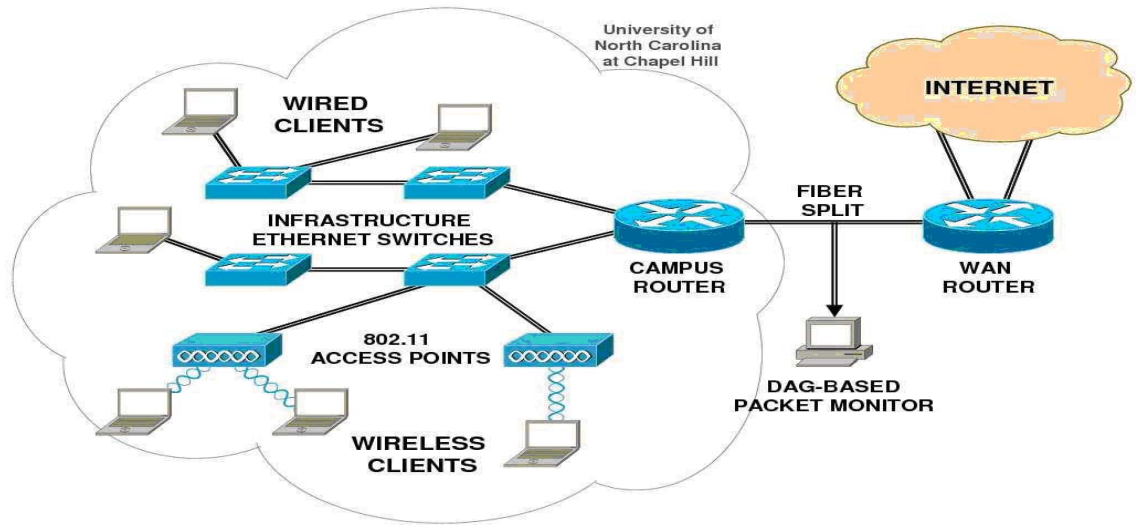


Figure 3.4: Monitor point for packet header collection for wired and wireless VLANs of UNC campus

For that reason, we selected to collect only packet headers from the largest wired and wireless VLANs of the UNC infrastructure. For the case of HSWN we captured only packets that crossed the gateway access node to the Internet. We thus did not capture any intra-WLAN traffic for any of the two infrastructures. As figure 3.4 shows, we placed the monitor at the egress router of UNC. Using a high precision DAG card, we sniffed every packet noticed on the link between UNC’s egress router and the rest of the Internet on both directions. On the same way, the monitor point for the case of HSWN was placed on the link between the gateway AN and the rest of the Internet.

### 3.4 Datasets acquired

Table 3.1 summarizes the data traces that are available for each infrastructure along with the tracing period they cover and their total size. For both wireless

	<b>UNC Wireless</b>	<b>UNC Wired</b>	<b>HSWN</b>
<b>Aggr/Client SNMP</b>	Sep 29 - Dec 31, 2004	N/A	Jun 26 - Jul 6, 2007
	Jan 1 - Jun 6, 2005		
	Jan 1 - Jun 30, 2006		
<b>Syslog</b>	Oct 1 - Nov 4, 2004	N/A	Jun 26 - Jul 6, 2007
	Jan 20 - Jun 26, 2005		
	Jan 1 - Aug 28, 2006		
<b>Packet headers</b>	Apr 13 - Apr 20, 2005	Apr 13 - Apr 20, 2005	Jun 26 - Jul 6, 2007
	Apr 28 - May 6, 2006		

Table 3.1: Summary of all traces collected from all infrastructures

infrastructures we have collected a rich set of different traces using all available monitoring mechanisms (we have also collected signal strength traces but they are not relevant with this study). These datasets, especially for the case of the UNC wireless infrastructure, cover a large period in time starting of from 2004 and ending in 2006, proving valuable in studies that focus on the evolution of a WLAN. Each one of the datasets acquired provides insight to different aspects of the client’s interaction with the infrastructure and is utilized in a different part of the traffic characterization process.

A first stage analysis of the traces collected can provide some additional characteristics of the infrastructures being monitored, such as, their sizes and number of users. Table 3.2 summarizes the number of distinct MAC addresses and number of APs that appear in the SNMP and Syslog traces collected from the wireless infrastructure of UNC. The corresponding number of distinct MAC addresses that appear in the SNMP and Syslog traces of the HSWN is 361. Moreover, based on the DHCP prefixes that each wireless and wired VLAN is bound to, we were

	Syslog		Aggr AP SNMP		Client SNMP	
	2005	2006	2004	2005	2004	2005
<b>Number of distinct MAC addresses</b>	21,046	26,612	N/A	N/A	16,018	20,422
<b>Number of distinct APs</b>	650	741	488	571	454	619
<b>Trace size</b>	3.78GB	2.63GB	1.72GB	1.51GB	1.44GB	3.38GB

Table 3.2: Number of distinct MAC addresses and APs per trace

Number of	UNC WLAN 2005	UNC WLAN 2006	UNC wired 2005	HSWN
<b>LAN side IPs</b>	8,196	7,848	3,919	1,663
<b>WAN side IPs</b>	2,979,365	2,491,219	2,056,926	7,584,951

Table 3.3: Number of LAN and WAN side IPs for HWSN and UNC infrastructures per tracing period

able to categorize all IP addresses that appear in the packet header trace into LAN (wired or wireless) and WAN side IPs. Table 3.3 summarizes the number of LAN side and WAN side IPs that appear in each packet header trace.

## CHAPTER 4

### Wireless traffic characterization and short-term forecasting

The main goals of this section is to characterize wireless traffic at the IEEE 802.11 MAC layer and identify statistical properties that will be used to enable APs to perform short-term traffic forecasting. Efficient forecasting algorithms can be beneficial in a wide range of mechanisms that aim at improving the utilization of 802.11 wireless networks, such as, load balancing, admission control, and QoS support. Forecasting algorithms provide traffic estimates that can be exploited to decide whether to accept a new association at an AP or redirect a client at a less overloaded AP. Traffic models that support forecasting algorithms can also be used to detect or even prevent abnormal user behaviors that would indicate malicious attacks or misconfigured devices. We also explore how short-term forecasting performs at two different spatial scales, namely client and AP.

The main contributions of this section are the following: We found that distribution of traffic across APs is rather skewed and follows a lognormal distribution. Moreover, some APs appear to be significantly more popular than others having been visited by nearly all clients of the wireless infrastructure. Our traffic load analysis also reveals a strong correlation in the log-log scale between total traffic and number of distinct clients seen at an AP. Analyzing the time series of traffic of the most utilized APs of the infrastructure reveals strong diurnal periodicities.

Based on these periodicities and recent history of an AP’s utilization we propose several models for wireless traffic demand. We employ these models to implement some simple short-term traffic forecasting algorithms and finally evaluate their performance.

The datasets used in this section are Aggregate, Client SNMP, and Syslog messages derived from the UNC campus-wide WLAN.

## 4.1 Traffic load notation

For the rest of the section we will often refer to the time series of traffic load of APs. Based on the SNMP trace for each AP, we produce a time series of its traffic load at hourly time intervals. This traffic is the total amount of bytes received and sent from all clients that were associated with the AP at that time interval. Based on the problem where traffic timeseries are used,  $T_i(h, d)$  will denote the traffic of AP  $i$  during the  $i$ -th hour of day  $d$ , while  $X_i(t)$  will denote total traffic of AP  $i$  observed during the  $t$ -th hour of the trace

### 4.1.1 Time series extraction

While our monitoring system requested traffic load information from each access point precisely every five minutes, missing values are relatively frequent in our dataset. The phenomenon is observed due to several reasons:

1. an AP may be down for maintenance, or in the middle of an accidental reboot;
2. an AP may be too busy to reply to an SNMP query;
3. the network path between our monitor and the AP may be temporarily

broken;

4. query packets and response packets may be lost (they are transported using UDP);

While these pathologies are expected to be infrequent, our dataset is large enough to contain numerous instances of each of them. Thanks to the cumulative nature of SNMP counters, we were able to reconstruct missing values quite accurately.

The basic technique for extracting an equally-spaced time-series  $X = \{x_1, x_2, \dots, x_n\}$  from SNMP data is to subtract the cumulative counters from two consecutive *polling* operations. In order to detect missing values and reboots, our polling samples include not only the cumulative counters but also the time of each polling operation, and the cumulative time that the access point has been running since the last reboot (*up time*). This means that the  $i$ -th polling sample for an access point has the form  $(t_i, u_i, c_i)$ , where  $t_i$  is time of the polling operation,  $u_i$  is the cumulative up time, and  $c_i$  is some cumulative counter (*i.e.*, total load in bytes). Given two consecutive polling samples, the load  $x_i$  observed between  $t_{i-1}$  and  $t_i$  is generally equal to  $c_i - c_{i-1}$ . There are two exceptions. First, SNMP counters are represented using 32 bits, so counters often wrap-around. We consider that a counter has wrapped around whenever  $c_i < 2^{30}$  and  $c_{i-1} > 3 * 2^{30}$ . In this case,  $x_i$  is equal to  $c_i + (2^{32} - 1 - c_{i-1})$ . Secondly, after a reboot, all the counters in an AP are reset. Therefore, if a reboot occurs at some point between  $t_{i-1}$  and  $t_i$ ,  $x_i$  is equal to  $c_i$  and the value of  $c_{i-1}$  should not be subtracted from  $c_i$ . Reboots can be detected by checking the value  $u_i$  in each polling sample. If  $u_i$  is significantly less than  $t_i - t_{i-1}$ , the access point has been reset, and  $x_i$  is equal to  $c_i$ . Otherwise,  $x_i$  is equal to the subtraction of the two cumulative counters. Note that resets may create situations that look like a wrap-around, so the detection of the reboots



should be performed before the detection of the wrap-arounds.

When all of the polling operations are successful,  $t_i - t_{i-1}$  is equal to the polling interval (*i.e.*, 5 minutes). However, when a polling operation fails,  $t_i - t_{i-1}$  is a multiple of the polling interval. If this is due to an access point reboot, the counter  $c_i$  only reports on the activity since the reboot operation. Therefore,  $c_i$  becomes the last value of the time-series. The values between  $t_{i-1}$  and  $t_i$ , for which no polling samples were available, are set to zero (access points have no load while off-line). If no reboot took place, the  $c_i - c_{i-1}$  does not correspond to a single  $x_i$  but to the  $m$  values of the time-series between  $t_{i-1}$  and  $t_i$ . In this case, we perform linear interpolation and set each intermediate value of the time-series to  $(c_i - c_{i-1})/m$ . Finally, note that  $t_i - t_{i-1}$  is not always exactly equal to the polling interval (or a multiple of it). The most significant cause is the retransmission mechanism in our SNMP monitor, which retransmits unanswered requests up to three times. Each new request is spaced by 5 seconds. Therefore, the maximum deviation of  $t_i - t_{i-1}$  with respect to the polling interval is 20 seconds, and our time-series extraction program takes into account this deviation.

## 4.2 Hotspot APs and their spatial locality

For several traffic characterization and modeling issues addressed in this study, we would like to focus our analysis on the most heavily APs. We do not need for example to use a forecasting algorithm to support load balancing on an AP that exhibits rather poor utilization. Instead, we are interested in APs that appear to be overloaded in terms of traffic.

For that, we define the *hotspots* of the wireless infrastructure based on three metrics:

- maximum hourly traffic;
- total traffic;
- maximum daily traffic;

*Hotspots based on maximum hourly traffic (set 1)*

These are the top  $\alpha\%$  APs ordered by their maximum traffic during an hour in the entire tracing period.

*Hotspots based on total traffic (set 2)*

These are the top  $\alpha\%$  APs ordered by their total traffic during the tracing period.

*Hotspots based on maximum daily traffic (set 3)*

These are the top  $\alpha\%$  APs ordered by their maximum traffic during a day in the entire tracing period.

#### 4.2.1 Hotspot definition

We define as a *hotspot* an AP that belongs in the top  $\alpha\%$  of APs with the highest maximum hourly traffic and in the top  $\alpha\%$  of APs with either the highest total traffic load or the highest maximum daily traffic load (i.e., the set  $(set1 \cap (set2 \cup set3))$ ). We will use this definition in the following sections.

We first investigate the spatial locality of the hotspots and name two APs co-located, if they are placed in the same building. How likely is to find co-located hotspots in the campus? We found that for  $\alpha=20$ , the percentage of co-located hotspots is above 76% and 79% for the hourly and total-traffic based definitions, respectively. 62% of the co-located APs belong in the  $(set1 \cap (set2 \cup set3))$ . For  $\alpha = 10$ , the corresponding percentages are about 11% smaller than their respective values for  $\alpha = 20$ . Note that, if using the uniform distribution, we had

randomly selected the same number of APs, the mean percentage of co-located APs in those selections is 48%.

### 4.3 Basic wireless traffic characterization

Whereas there is rich literature on traffic characterization in wired networks, (e.g, [83, 76, 73, 71, 85]), there is significantly less work on the same detail for WLANs. Using aggregate and client SNMP we explore various aspects of wireless traffic demand, such as, distribution of traffic across APs, number of distinct clients and associations per AP and correlation between number of clients and total traffic at various scales.

#### 4.3.1 Traffic and client distribution across APs

Figure 4.1 presents the total traffic that each AP has transferred during the monitoring period measured using SNMP data. As this plot shows, traffic distribution across APs is highly skewed (note that y-axis is in the log-scale). The most heavily utilized APs appear to have transferred nearly 250GBytes or more while the least utilized ones have transferred just some KBytes during the same tracing period. Looking more carefully at 4.1, especially for ranks of APs between 75 and 300 approximately, we can notice a clear linear trend in the traffic among APs. This suggests that total traffic for the majority of the APs can be modeled using an exponential model of the form:  $10^{a*x+b}$ . The parameters  $a$ ,  $b$  of the model were computed using linear regression between empirical data and theoretical model resulting in an coefficient of determination -  $R^2$  value equal to 0.9925 suggesting a very accurate fit. The suggested model fails however to capture the “heavy-tail” trend observed in the distribution of total traffic across

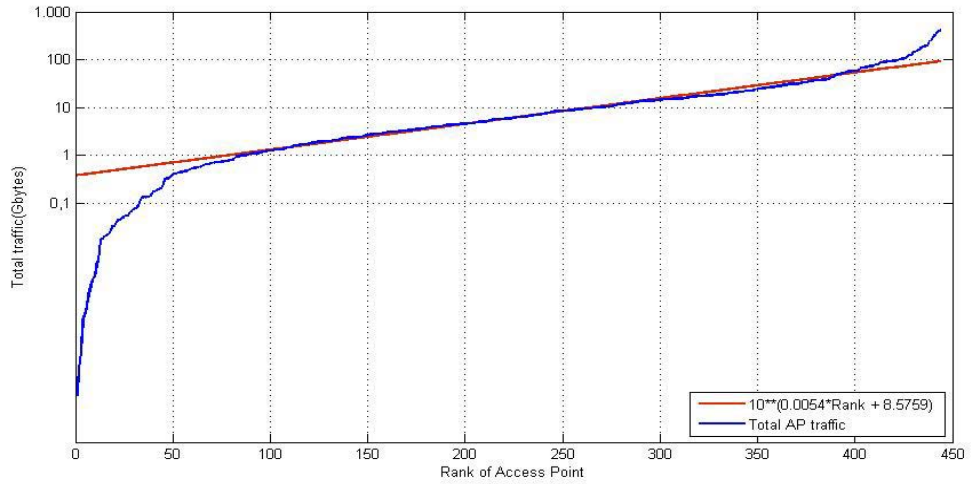


Figure 4.1: Distribution of total traffic (GBytes) across APs

APs. To be more precise, for 50 APs the actual utilization is lower than the theoretical one while for the top 50 APs in terms of total traffic, the proposed model underestimates their total load.

Driven by the clear log-linear trend observed in 4.1, we drew the Complementary Cumulative Distribution Function of total traffic across APs which suggested a possible lognormal model. Using Maximum Likelihood Estimate, we computed the parameters of the lognormal distribution which were  $\mu = 22.1669$  and  $\sigma = 2.4625$ . Figure 4.2 shows the quantile-quantile plot of the total traffic per AP against the theoretical quantiles of a log-normal distribution with parameters  $\mu = 22.1669$  and  $\sigma = 2.4625$ . The quantiles for the vast majority of traffic values fall well within the confidence interval with the exception of APs which transfer insignificant amounts of traffic.

Another metric for the utilization of APs apart from their total traffic is the average throughput observed during the tracing period. Based on the total traffic

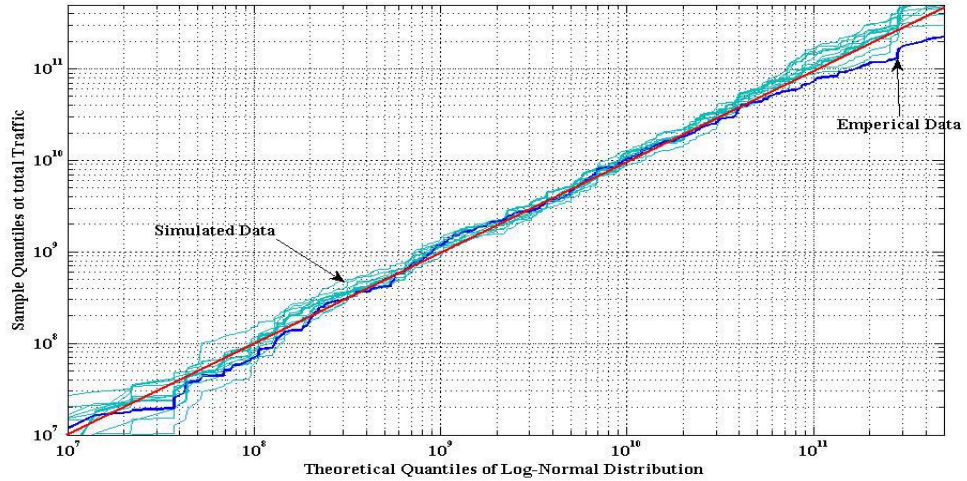


Figure 4.2: Quantile-quantile plot for total traffic per AP

transferred through each AP during each hour of the trace, we compute average hourly utilization. For APs running IEEE802.11b the maximum throughput observed during one hour of the trace is approximately 4.3Mbps while for APs running IEEE802.11g is 24MBps. Figure 4.3 is the CCDF of the average hourly throughput for each AP. It is interesting to note that average utilization per hour, even for the most heavily loaded APs, rarely exceeds 0.1Mbps which is far beyond the theoretical throughput limit for IEEE802.11b. The reason for such a low average throughput is that AP traffic within an hour is highly variable exhibiting both large peaks and idle 5 minutes intervals during which no traffic is transferred.

In order to identify the most popular APs we used Syslog messages to derive the number of distinct clients that have visited each AP along with the total number of associations per AP throughout the tracing period. Figure 4.4 presents the number of distinct clients that have visited at least once each AP.

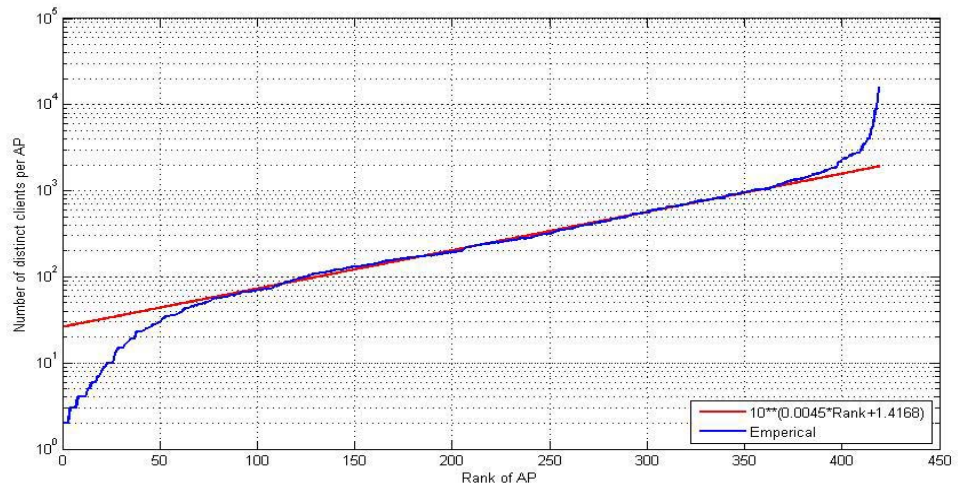


Figure 4.3: CCDF of average hourly utilization per AP

A visit of client 'C' at AP 'i' at time  $t_i$  is recorded if we notice a Syslog message of type “Associated” or “Reassociated” from AP 'i' at  $t_i$  without having received a “Dissassociated” message of the corresponding client through that AP the same time instance. The most popular APs of the campus have been visited by nearly 16,000 distinct clients. As in the case of total traffic, figure 4.4 shows that there is a clear log-linear trend between AP ranks 75 and 375 approximately. Fitting the total number of distinct clients per AP with an exponential model of the form:  $10^{\alpha*x+\beta}$ , through linear regression results in a coefficient of determination of  $R^2=0.0975$ . As the corresponding figure shows, while the exponential model provides an accurate fit for the majority of the APs, it fails to capture the heavy-tail trend observed for AP ranks smaller than 75 and larger than 375. The log-linear trend observed in the figure 4.4 and the heavy-tailed shape of the CCDF total number of clients having visited each AP suggests a possible fit through a lognormal distribution for the number of distinct clients per AP. The parameters of the lognormal model computed using MLE over the number of distinct clients

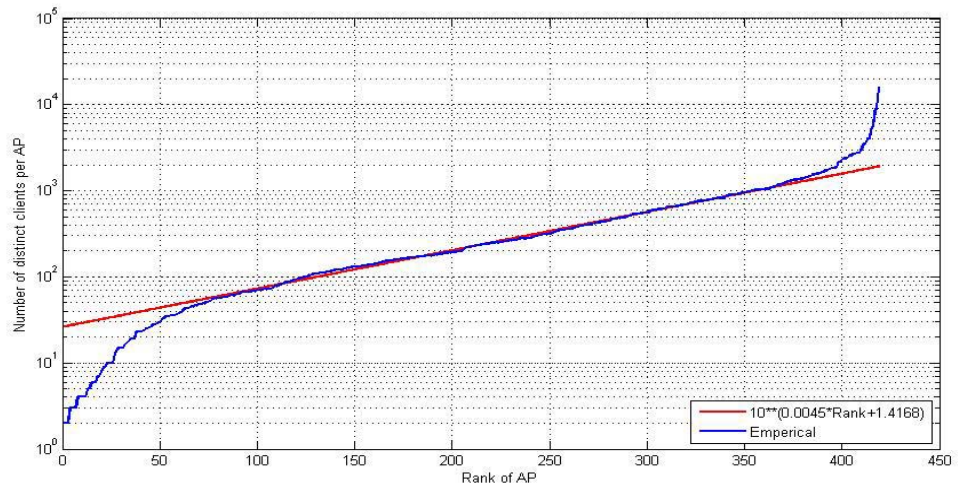


Figure 4.4: Number of distinct clients per AP

per AP were  $\mu = 22.1669$  and  $\sigma = 2.4625$ .

A complementary metric for the popularity of APs is the total number of visits that each one has seen. Consider the case of a client that visits AP 'i' and then gets disconnected. For the rest of its wireless life, this client associates periodically at AP 'j' to exchange mail or visit Web pages for example. In that case, although both APs have been visited by the same number of distinct clients, AP 'j' has received significantly more visits. Apart from the popularity of APs, the number of visits observed at each AP can be an indication of “hot” areas in the campus or of possible dead-spots. APs that have received a significant number of visits may reside in “hot” areas of the infrastructure like lecture halls in academic buildings while APs with few visits could indicate APs located in remote buildings or in areas with weak coverage which require placing more APs or increasing the transmission range of current APs. Figure 4.5 is a CCDF plot of number of visits per AP.

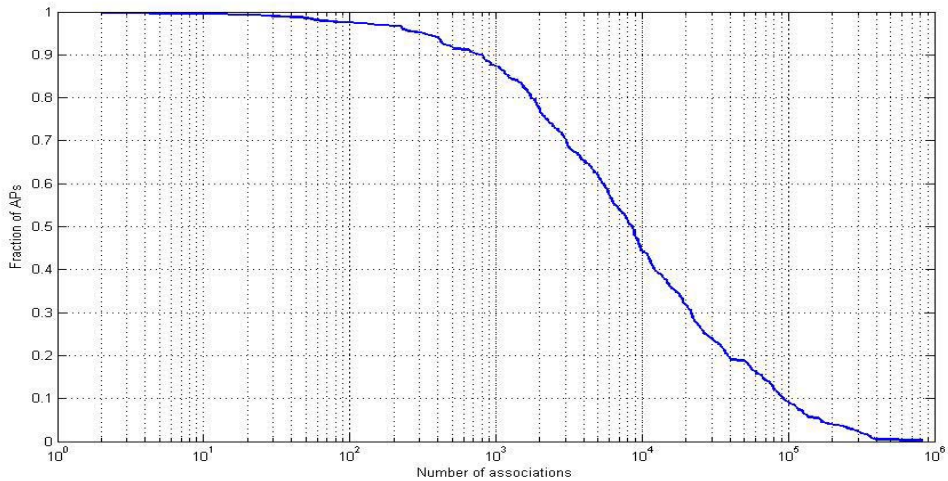


Figure 4.5: CCDF of number of visits per AP

As this plot shows, the distribution of number of visits per AP is highly skewed. While 90% of the APs have seen more than approximately 1000 visits, only 10% of them have been visited more than 10,000 times which constitutes an average of approximately 600 visits per day.

#### 4.3.2 Correlation between total traffic and number of distinct clients per AP

One of the phenomena that we wanted to explore, as part of our wireless traffic characterization effort, is whether there is any correlation between number of distinct clients and total traffic on an AP. Previous studies [61], have shown that there is a weak correlation between offered load at APs and number of distinct clients. Authors in [70] have shown that offered load is more sensitive to individual client traffic characteristics rather than just the number of clients. Our assumption is that total traffic load observed at an AP should be affected



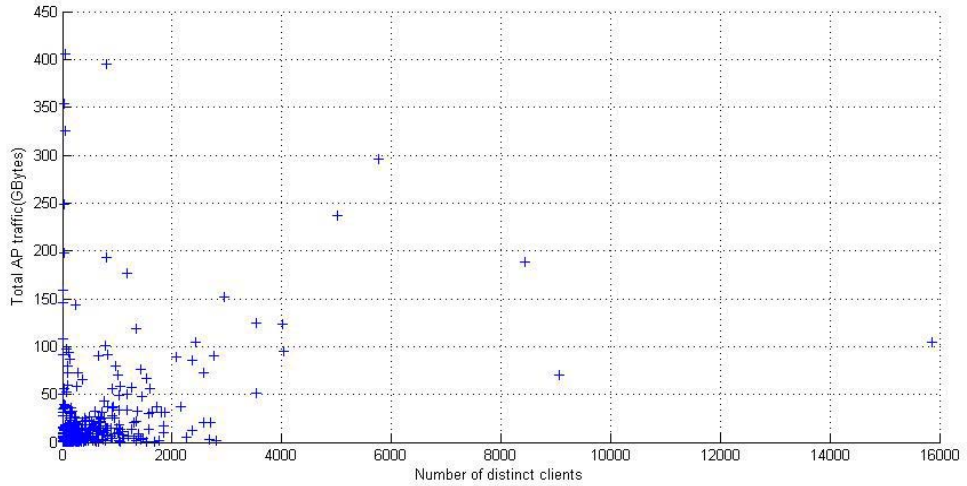


Figure 4.6: Total AP traffic vs. number of distinct clients - linear scale

the by number of distinct clients that have visited this AP. Figure 4.6 is a scatter plot of total traffic against number of distinct clients per AP. As this plot shows, there is no clear trend in the linear scale suggesting a weak correlation between offered load and number of distinct clients at APs. Looking however at the same plot at the log-log scale (figure 4.7), we notice that there is a clear upwarding trend that can be expressed through the following model  $\log_{10}tt = a * \log_{10}dc + b$  where  $tt$  is the total traffic per AP and  $dc$  the number of distinct clients per AP. This means that total traffic  $tt$  of an AP can be expressed as:  $tt = 10^b * dc^a$ . Using linear regression we found that  $a = 0.57$  and  $b = 8.38$ .

In order to explore the correlation between total traffic and number of distinct clients per AP, we extracted for all hotspot APs the time series of total traffic and number of distinct clients per hour. Then, the cross-correlation coefficient was computed between the time series of total traffic and number of distinct clients for each AP at various scales. Note that when cross-correlation is run over 2 time series of  $n$  values, the correlation coefficient is computed between

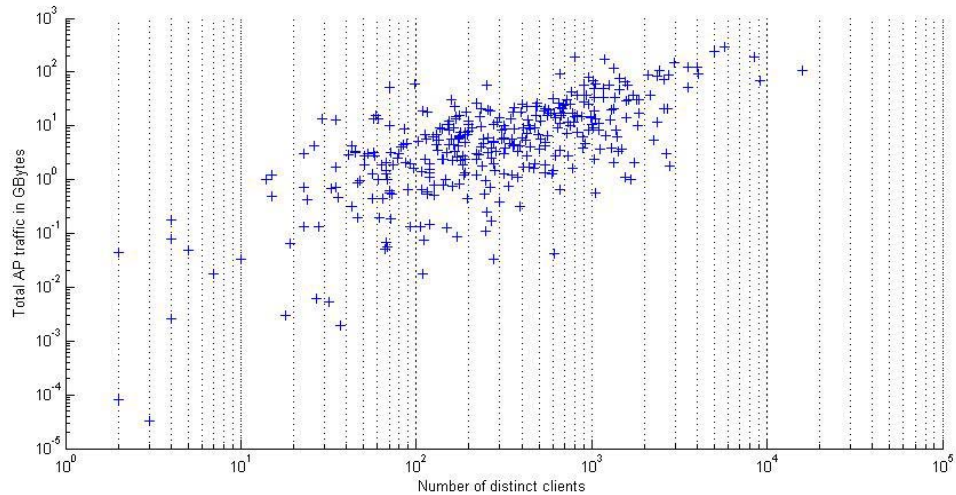


Figure 4.7: Total AP traffic vs. number of distinct clients - log scale

the two timeseries at all possible  $2*n-1$  lags. Table 4.1 summarizes the cross-correlation coefficients at lag 0 for 5 randomly selected hotspots along with the average coefficient among all hotspots.

Note that the two last columns of the table contain the cross-correlation coefficient between timeseries of total traffic and number of distinct APs per hour normalized using the log-scale and the  $\sqrt[4]{}$  scale. As this table shows, although there is a weak correlation between total traffic and number of distinct clients per hour in the linear scale for most hotspots, the correlation coefficient is above 85% for all hotspots in the two normalized scales used. The average cross-correlation coefficient among all hotspots in the log-scale is 88.21% certifying the upwarding trend noticed in 4.7.

As this analysis has shown, as the number of distinct clients grows, there is an exponential increase in the observed traffic load. This observation can prove valuable to algorithms and mechanisms that wish to perform load balancing in

Hotspot ID(i)	$X_i(t) - DC_i(t)$	$\log(X_i(t)) - \log(DC_i(t))$	$\sqrt[4]{X_i(t)} - \sqrt[4]{DC_i(t)}$
<b>222</b>	55.52%	89.71%	95.46%
<b>400</b>	65.13%	99.28%	96.67%
<b>404</b>	27.18%	95.23%	91.27%
<b>406</b>	53.0%	97.66%	89.68%
<b>472</b>	76.13%	84.69%	96.67%
<b>Average</b>	46.94%	88.21%	90.63%

Table 4.1: Cross-correlation coefficients between number of distinct clients and total traffic APs where a large number of clients is either observed or is expected to arrive during the following time intervals. The observations of this analysis can also be applied to support and enhance capacity planning by placing more APs in over-crowded areas in order to minimize the load of current APs.

#### 4.4 Wireless traffic modeling and forecasting methodology

This section describes three categories of forecasting algorithms that are based on simple models for wireless traffic. Based on periodicities detected in the time series of traffic of APs we build algorithms that exploit such diurnal or weekly patterns. The second category of forecasting algorithms exploits temporal dependencies that appear in traffic time series of AP. These algorithms take into account a *window of recent-traffic history* (e.g the traffic observed during the last three hours). When observed traffic load at APs exhibits both strong periodicities and temporal dependencies, hybrid algorithms can consist a better choice.

Our general methodology consists of the following steps:

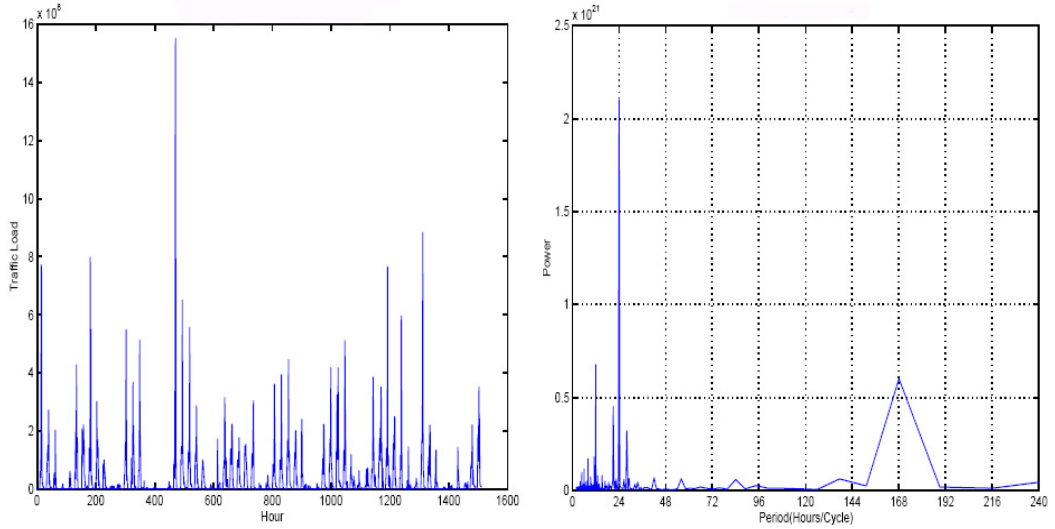


Figure 4.8: Traffic load at AP 472: (a) time series (b)Power spectrum

- (a) Time-series extraction, data cleaning, and treatment of missing values (addressed in 4.1.1);
- (b) Power spectrum and partial autocorrelation analysis;
- (c) Traffic load modeling; and
- (d) Forecasting using the traffic load models.

#### 4.4.1 Spectrum analysis

We find that the aggregate hourly traffic for all APs in the infrastructure exhibits diurnal and weekly periodicities. Similar trends are observed in the hourly traffic for several APs by autocorrelation plot and spectrum analysis. 10 out of the 19 hotspots have a clear spike at 24 hours/cycle and do not have a high frequency variation. Also, some APs have weekly patterns at around 168 hours/cycle.

Figure 4.8(a) and (b) show the time series and spectrum plots of the hotspot

AP 472. This AP exhibits strong diurnal periodicity. There are other APs with no clear periodic pattern, for which there is little prediction power among the historical data. Further smoothing does not appear to be helpful, at least with our current relatively short traces.

#### 4.4.2 Forecasting algorithms based on wireless traffic models

##### 4.4.2.1 Forecasting using historical means and recent traffic

First, we model the traffic load at an AP during an hour. The model facilitates the diurnal and weekly periodicity of the traffic load. We define the *historical mean hour* traffic of an AP as the mean of the traffic during that hour for each day in the history of that AP ( $N_{days}$  days). We only consider weekdays. For example, the historical mean-hour traffic for AP  $i$  is defined as

$$\mu_i(h) = (1/N_{weekdays}) \times \sum_{d=1}^{N_{days}} T_i(h, d) * IsAWeekday?(d),$$

where  $h = 1, \dots, 24$  and  $IsAWeekday?(d)$  is a binary indicator function that specifies whether or not the  $d$ -th day is a weekday, and

$$N_{weekdays} = \sum_{d=1}^{N_{days}} IsAWeekday?(d).$$

Similarly, the *historical mean hour-of-day* traffic is the mean of the traffic at such hour of day in the history of that AP. For example, the mean hour-of-day for AP  $i$  is defined as

$$\mu_i(h, l) = (1/nw(l)) \times \sum_{k=1}^{N_{days}} IsWeekday?(k, l) \times T_i(h, k),$$

where  $h = 1, \dots, 24$ ,  $l$  “runs” from “Mon” through “Sun”, and

$$nw(l) = \sum_{k=1}^{N_{days}} IsWeekday?(k, l).$$

The  $IsWeekday?(x, l)$  is a binary indicator function that specifies whether or not the  $x$  is a weekday  $l$ . The  $nw(l)$  counts the total number of weekdays  $l$  (e.g., the number of Mondays). For example, for the  $\mu_i(2)$ , we take the historical mean of the traffic at AP  $i$  for all days in the history at 2am. Similarly, for the  $\mu_i(2, "Mon")$ , we compute the mean of the traffic of all Mondays at 2am.

We Taylor two simple models based on the historical mean hour and mean hour-of-day. Specifically, for each AP (e.g., AP  $i$ ), we define the models  $Z_i^1$  and  $Z_i^2$ , as follows:

$$(P1) \quad Z_i^1(h, d) = \mu_i(h)$$

$$(P2) \quad Z_i^2(h, d) = \sum_{l \in \{Mon, \dots, Sun\}} IsWeekday?(d, l) \times \mu_i(h, l).$$

To incorporate the recent traffic information in the traffic model, we compute the mean traffic during the last  $w$  hours. For each AP (e.g., AP  $i$ ), we introduce the *weighted average of the recent traffic mean and the historical mean hour and hour-of-day*,  $Z_i^3$  defined as

$$(P3) \quad Z_i^3(h, d) = a \times (1/w) \sum_{k=t-w}^{t-1} X(k) + b \times \mu_i(h, d) + c \times \mu_i(h).$$

We experiment with different window sizes and weights to evaluate the impact of the recent history and periodicity on forecasting. Note that the P3 with weights (a,b,c) equal to (1,0,0) and history window  $w$  has the form of an autoregressive process of order  $w$ ,  $AR(w)$ . In that case, the prediction takes into account only the recent traffic history instead of the periodicity. We can specify the weights of the P3 using multiple linear regression. The purpose of the multiple linear regression is to establish the relationship among the group of predictors, namely, the history window, historical mean hour traffic, and the historical mean hour-of-day. This allows us to understand which predictors have the greatest effect. The

linear model takes the form  $y = Xb + e$ , where  $y$  is a vector of observations,  $X$  is a matrix of independent variables (regressors/predictors) and  $e$  is a vector of random disturbances. Multiple linear regression aims to obtain the best fitting curve by minimizing the least square errors ( $\sum_{i=1}^n [y - f(X_i)]^2 = \sum_{i=1}^n [y_i - (bX_i)]^2$ ). P3 with weights defined using multiple linear regression is denoted as P3-MLR. We propose three simple prediction algorithms based on the aforementioned models. P1 and P2 use the historical means to compute the  $Z_i^k(h, d)$ ,  $k = 1, 2$  for P1 and P2, respectively, and predict the traffic load of AP  $i$  during the  $t$ -th time interval (that corresponds to the  $h$ -hour of day  $d$ ). P3 integrates the historical means of hour and hour-of-day with the recent traffic history. More specifically, P3 is an *one-step ahead prediction algorithm*, since for the recent traffic, it uses the *actual* traffic values as opposed to the predicted ones (for the next-hour prediction).

## 4.5 Evaluation of the performance of the forecasting algorithms

This section introduces two metrics for the evaluation of the proposed forecasting algorithms and performs a comparative analysis of their performance.

### 4.5.1 Metrics: prediction error ratio and percentage of correct predictions

To evaluate the performance of the prediction algorithms, we compute the *prediction error ratio* which is the ratio of the absolute difference of the predicted from the actual traffic over the actual traffic ( $r$ ). For the prediction of the traffic of AP  $i$  at time  $t$ , the prediction error ratio  $r(t)$  is defined as  $r(t) = |Z_i^k(t) - X_i(t)|/X_i(t)$ , for prediction algorithms  $k = 1, 2, 3$ . A perfect prediction algorithm has predic-

tion error ratio equal to 0.

Note that the way we defined the prediction error ratio represents the percentage by which the corresponding forecasting algorithm underestimates or overestimates the actual traffic. Consider for example the case where the actual traffic is 50KBytes while the corresponding forecasting algorithm estimates next's hours traffic for a specific AP to be 25KBytes. The value of the prediction error ratio is computed as 0.5 indicating that we underestimate actual traffic by 50%.

The prediction algorithms apply a predicted interval based on the historical mean and a tolerance (or precision) error level. Specifically, we define the  $\epsilon$ -tolerance prediction interval from a mean  $\mu$  to be the interval  $[(1-\epsilon)*\mu, (1+\epsilon)*\mu]$ . The prediction algorithm computes the percentage of times that the actual traffic is in the predicted interval. For example, in the case of the prediction  $P_k$ ,  $k = 1, 2, 3$ , for the traffic of AP  $i$  during the  $h$ -th hour of day  $d$ , it computes the prediction interval

$$[(1 - \epsilon) * Z_i^k(h, d), (1 + \epsilon) * Z_i^k(h, d)],$$

and checks if  $X_i(t)$  is in that interval.

A good prediction algorithm should have a high correct prediction percentage and low prediction error ratio. A large prediction error ratio indicates large prediction estimates and may result in conservative prediction and resource underutilization.

#### **4.5.2 Forecasting using historical means and recent traffic (P1, P2, P3)**

For all the aforementioned prediction algorithms, we computed the means based on the history for each AP. The history corresponds to three weeks of the trace,



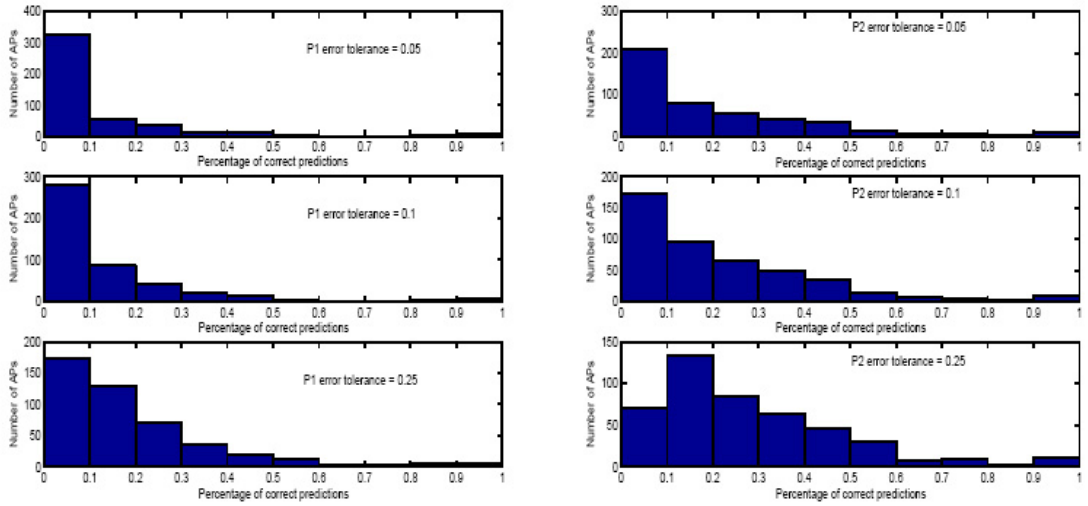


Figure 4.9: Performance of prediction algorithms P1, P2 considering all APs

excluding weekends and starting on Monday, October 18th, 2004. We predict the traffic for each AP, for all the hours during the weekdays of the following week (Monday, November 8th until Friday, November 12th). We call this period *forecasting period*. For P3, we varied the recent history window size to be 2, 3, 4, and 5 hours. We evaluated P3 for various values of  $a, b$ , and  $c$ , including also values resulted from applying multiple linear regression for each AP.

Figures 4.9 and 4.10 show the histograms of the percentage of correct predictions for the P1, P2, and P3 considering all APs. P3 outperforms P2 and P1 with respect to the correct predictions percentage. P3 also outperforms P2 and P1 with respect to the correct predictions percentage, when we only consider the hotspots. Specifically, for a window of two hours and  $(a, b, c)$  equal to  $(1, 0, 0)$ , P3's percentage of correct predictions for a 25%-tolerance prediction interval has a (mean, median, std. deviation) equal to  $(34.17\%, 24.17\%, 22.86\%)$ .

The *mean percentage of correct predictions* of hotspots for an  $\epsilon$ -tolerance is

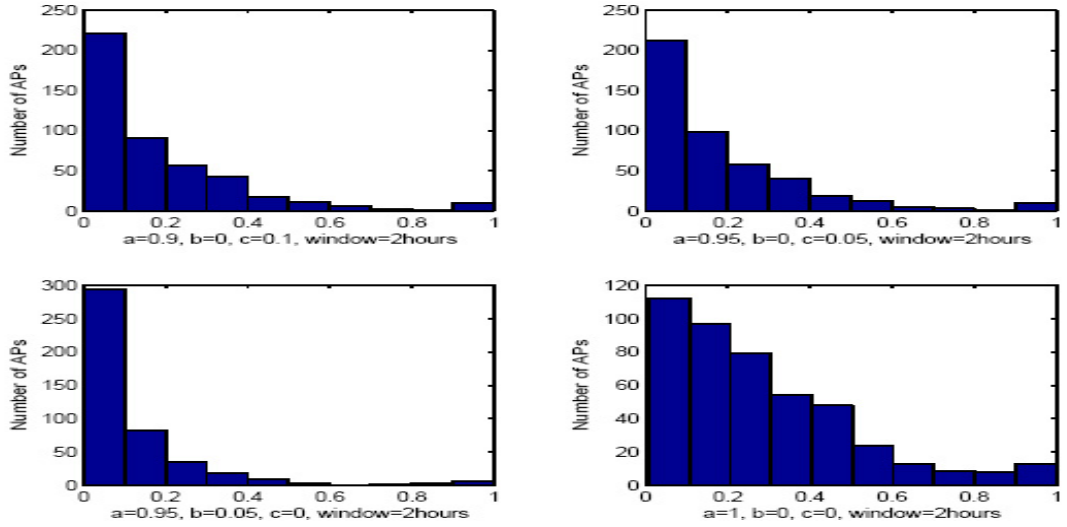


Figure 4.10: Performance of prediction algorithm P3 with 25% error tolerance considering all APs

the average of the percentages of correct predictions for that  $\epsilon$ -tolerance considering all hotspots. The *mean prediction error ratio* of hotspots is the average of the mean prediction error ratios considering all hotspots. In the same manner, we compute their median and std. deviation. For the same  $\epsilon$ -tolerance, P2 has a lower percentage of correct predictions than P3 but higher than P1 (for both median and mean prediction of correct percentages). Similarly, the median prediction error ratio for P3 is lower than for P1 and P2 (see Figure 4.12). On the other hand, P3's mean prediction error ratio is lower than P1's and higher than P2's one. The high mean prediction error ratio of P1, P2, and P3 are due to the high variability in the traffic (figure 4.11).

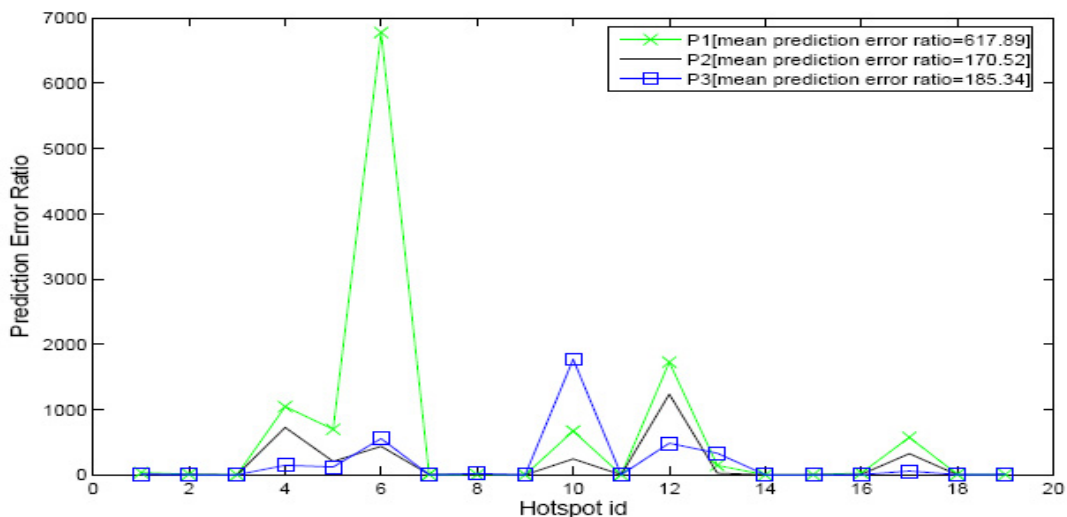


Figure 4.11: Mean prediction ratios for P1, P2, and P3 with weights  $(a,b,c)=(1,0,0)$  for each hotspot

## 4.6 Client level forecasting

In this section we perform traffic forecasting at the client level. We expect that client traffic will be more bursty and exhibit more temporal phenomena than AP traffic.

We repeated the same forecasting methodology that was used for APs (described in 4.4) and focused on clients that transfer 75% or more of their traffic through a specific building. We name this building as *home building* of that client. We first used spectrum analysis to identify clients that exhibit strong diurnal or weekly periodicities and found that 30.8% of them exhibited 24 hour periodicities. Figures 4.14(a) and 4.14(b) show the corresponding power spectrum for 2 of the most active clients during the tracing period studied.

Moreover, applying partial autocorrelation analysis to the normalized time-series of all periodic clients suggests that hourly traffic for the majority of them

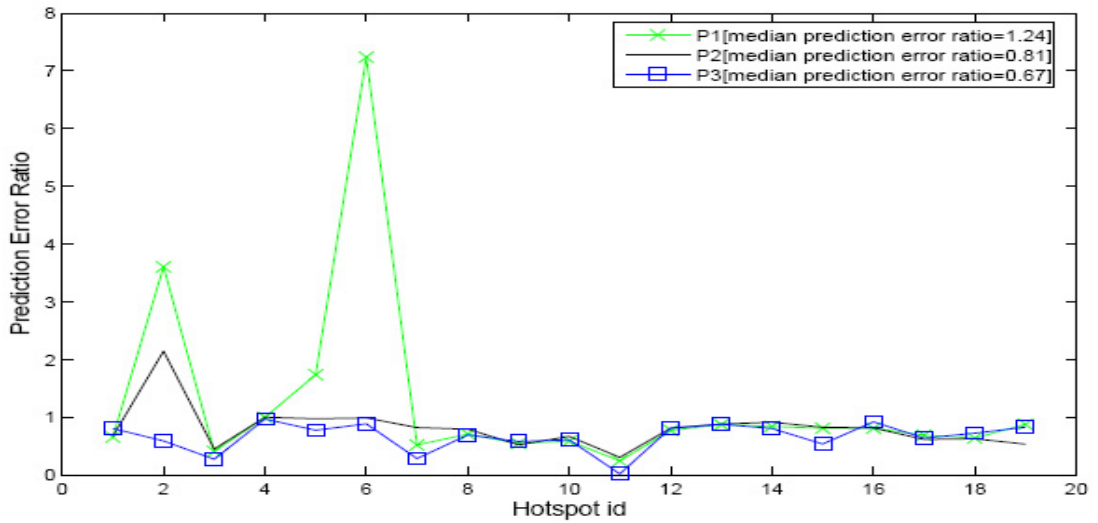


Figure 4.12: Median prediction ratio for P1, P2 and P3 with weights  $(a,b,c)=(1,0,0)$  for each hotspot

can be accurately characterized through an  $AR(1)$  model. Figures 4.14(a) and 4.14(b) present the partial autocorrelation plots for clients 199 and 373 respectively. Notice the sudden cut-off for both clients at lag 1 while all other coefficients fall within the confidence interval envelope.

We applied the historical means and recent history algorithms described in 4.2, using three weeks of history for all periodic clients. For recent history we varied the history window to be 1, 2, 3, and 5 hours while parameters  $a$ ,  $b$ , and  $c$  of the P2 algorithm, were computed using multiple-linear regression for each client. Table 4.2 summarizes the mean, median, and SD of the relative prediction error ratio for  $P1$ ,  $P2$ ,  $P3$  and recent history algorithms averaged over all periodic clients. Median relative error prediction for example, (as denoted in Table 4.2), is the average over each client’s median relative error ratio. Comparing the values of the relative prediction error ratio for various sizes of the recent history window, we found that a window of size one performs better.

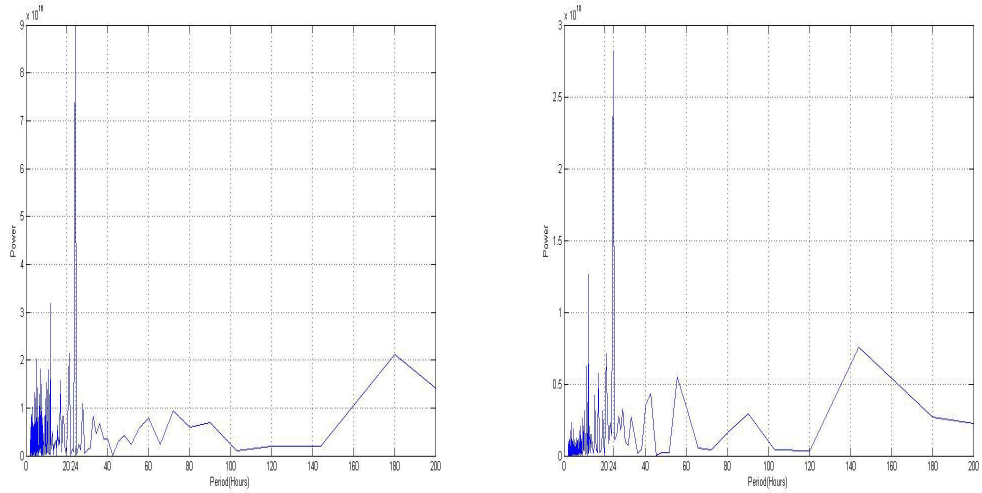


Figure 4.13: Power spectrum for clients (a) 199 (b) 373

	P1	P2	P3,window=1	Recent history,window=1
<b>Mean</b>	22.69	30.50	38.98	32.26
<b>Median</b>	3.57	6.25	4.02	10.38
<b>SD</b>	66.65	91.75	114.13	73.06

Table 4.2: Relative prediction error ratio for P1, P2 ,P3 and recent history algorithms for client traffic forecasting

As expected, P1 performs better in terms of mean, media, and SD of the error ratio since most periodic clients were found to exhibit 24 hours periodicities but not weekly ones. This is also obvious in the P3 algorithm which has a higher median error ratio than P1 but lower than P2. Overall, recent history performs worse in terms of mean and median than P1 and P2 while P1 which exploits the 24 hours periodicities performs better than all other algorithms. The reason why recent history performs worse than all periodic algorithms is that it fails to capture the periodic patterns that exist among client traffic timeseries. Note also

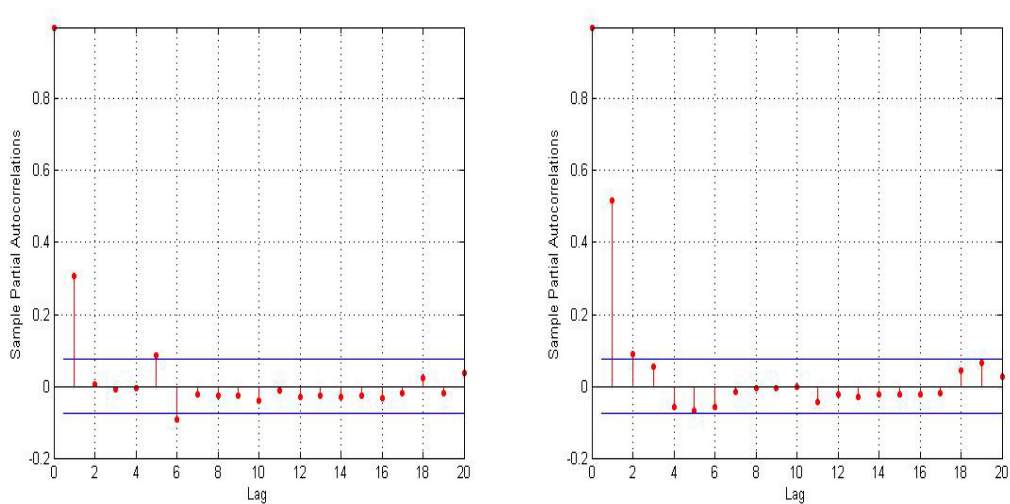


Figure 4.14: Partial Autocorrelation Function for clients (a) 199 (b) 373

that client traffic is rather bursty often exhibiting large peaks of traffic followed by periods during which clients transfer insignificant amounts of traffic.

Comparing the performance of the periodic-based and recent-history algorithms at the AP- and client-level, we notice that P1,P2, and P3 have a significantly lower mean relative prediction error ratio when used to forecast client traffic. Looking at plot 4.11 shows that although the mean relative error ratio is close to 1 for most of the hotspots, there are two of them for which the corresponding error is as high as 500. Apart from that, mean relative error ratio is lower when P1, P2, and P3 are applied to forecast client traffic due to a large number of clients that transfer insignificant amounts of traffic contributing thus with a small value to average error. On the other hand, median prediction error ratio is significantly lower in the case of AP traffic forecasting suggesting that P1, P2, and P3 perform Overallly better when used to forecast AP traffic.

## 4.7 Discussion

### 4.7.1 Other forecasting approaches

In this section we discuss other forecasting approaches that make use of complex statistical methods. The two forecasting methodologies discussed here are based on singular spectrum analysis and on  $p$ -order ARIMA models for traffic load.

#### 4.7.1.1 Forecasting based on Singular Spectrum Analysis (SSA)

A relevant, novel, forecasting methodology based on Singular Spectrum Analysis (SSA) is presented in [5]. SSA is a non-linear time-series analysis method that is used to reveal underlying patterns and structure in wireless traffic. Authors in [5] observe that time-series that correspond to wireless traffic are often short and contain peaks on top of a more regular background. Besides, these series often have both regular (periodic) and irregular (noisy) aspects which may be present in different spatial and temporal scales. Motivated by these observations, they use Singular Spectrum analysis to decompose traffic time-series into two components, namely, a low frequency one representing the main trend of the time series, and a high-frequency one capturing the noise of the time-series. This decomposition provided by SSA is incorporated in a forecasting algorithm which focuses on predicting traffic values in a relatively long-term horizon based on the component which captures the low-frequency behavior of the original time-series.

#### 4.7.1.2 Forecasting using ARIMA models

In this section we explore a short-term forecasting approach based on ARIMA models for traffic. Partial autocorrelation analysis for the traffic load time-series of the majority of the hotspots APs has revealed a significant peak on lag 1

indicating that an ARIMA model comprises a reasonable solution for AP traffic load. The first part of this section focuses on normalizing and applying some statistical analysis on normalized traffic load for hotspot APs. On the second part of the section, we implement and evaluate the performance of a forecasting algorithm based on the derived ARIMA model.

**Data normalization and traffic load modeling** As Figure 4.8(a) shows, total traffic per hour for AP 472 is highly variable and bursty exhibiting local spikes that are hard to predict. This pattern holds for the majority of the hotspots whose total traffic is not uniformly distributed throughout the tracing period. Figure 4.15(a) plots the normal quantile plot of  $X_{472}(t)$  for AP 472, which clearly suggests that the marginal distribution of the traffic load is heavily skewed to the right. This calls for a suitable transformation to make the data closer to a normal distribution. Such a transformation can reduce the effect of those local spikes on the forecasting performance. In addition, standard time series modeling procedures are most suitable for situations with normal data [82].

After experimenting with different transformations, the 1/4 power transformation,  $Y(t) = X_{472}^{1/4}(t)$ , seems to give the best result. In particular, Figure 4.15(b) gives the normal quantile plot for the transformed load  $Y(t)$  at AP 472. As one can see,  $Y(t)$  is much closer to be normally distributed, and does not have extreme outliers as those in Figure 4.15(a). The following model will be performed on  $Y(t)$ .

We first point out that  $Y(t)$  exhibits strong non-stationarity in both the mean and the variance. Figure 4.16(a) plots the bimodal changing patterns of its mean, median, 25-th percentile and 75-th percentile as functions of hour-of-day ( $h(t)$ ), which shows that both the mean and the percentiles change across the day. For



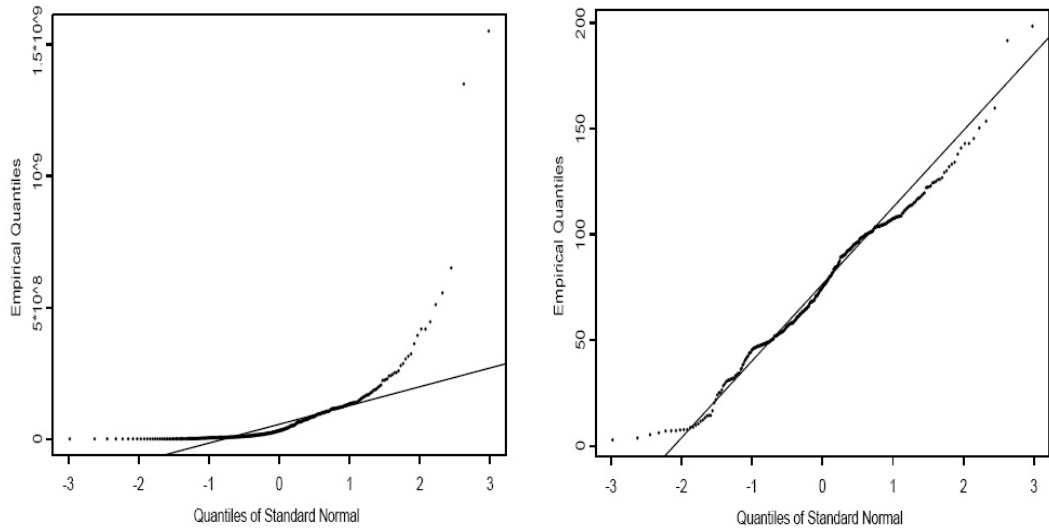


Figure 4.15: Traffic load at AP 472: (a) normal quantile plot for  $X_{472}(t)$  (b) normal quantile plot for  $Y(t)$

example, the mean curve suggests that there is very little traffic between midnight and 7-8AM; then the load starts to increase until it reaches the first mode around 10AM and stays flat until noon; after lunch-break, the load increases again to the second mode around 3PM before it starts to decrease until midnight. Very sensible explanations can be given for such a diurnal pattern.

Similarly, Figure 4.16(b) indicates the diurnal patterns for the standard deviation and Inter Quartile Range (IQR) (*i.e.*, the difference between the 25-th and 75-th percentiles). The plot suggests that there is increasing variability in the traffic load during 7AM-10AM and 1PM-3PM, exactly when the load increases. In addition, the variability stays small between 10AM and 1PM.

The above exploratory data analysis motivates us to normalize the transformed load  $Y(t)$  in the following way,

$$e(t) = \frac{Y(t) - \mu_{h(t)}}{\sigma_{h(t)}}$$

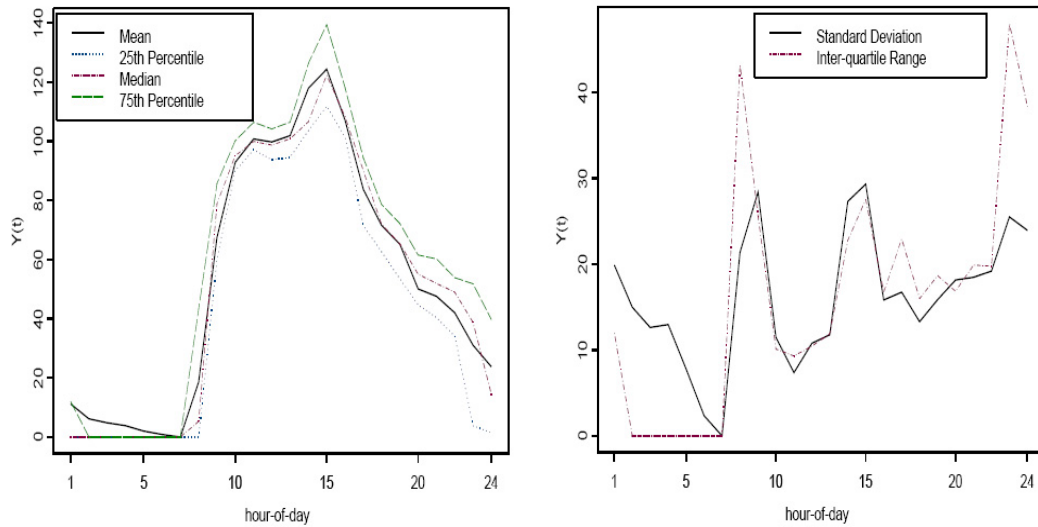


Figure 4.16: Changing patterns of: (a) mean, median, and quantiles of  $Y(t)$  (b) standard deviation (SD) and inter-quantile range of  $Y(t)$

where  $h(t)$  is the corresponding hour-of-day for time  $t$ ,  $\mu_{h(t)}$  is the mean of  $Y(t)$  during those time periods with the hour-of-day being  $h(t)$  while  $\sigma_{h(t)}$  is the standard deviation of  $Y(t)$  during those time periods, and  $e(t)$  can be treated as a normalized version of  $Y(t)$ . Note that  $\mu_{h(t)}$  and  $\sigma_{h(t)}$  have been plotted in Figure 4.16(a) & (b) for AP 472.

After the normalization, we can assume  $e(t)$  to be a stationary time series as shown in Figure 4.17(a). The corresponding partial autocorrelation function (Partial ACF) (Figure 4.17(b)) suggests that an AR(1) model is reasonable for the normalized time series,  $e(t)$ . Thus, we fit a family of AR( $p$ ) models to  $e(t)$  using the Yule-Walker method and select the approximate order  $p$  by minimizing the Akaike Information Criterion (AIC). See Brockwell and Davis (1998) for details about the estimation method and the model selection criterion, AIC. Note that the order  $p$  specifies the number of lagged variables in the time series model and

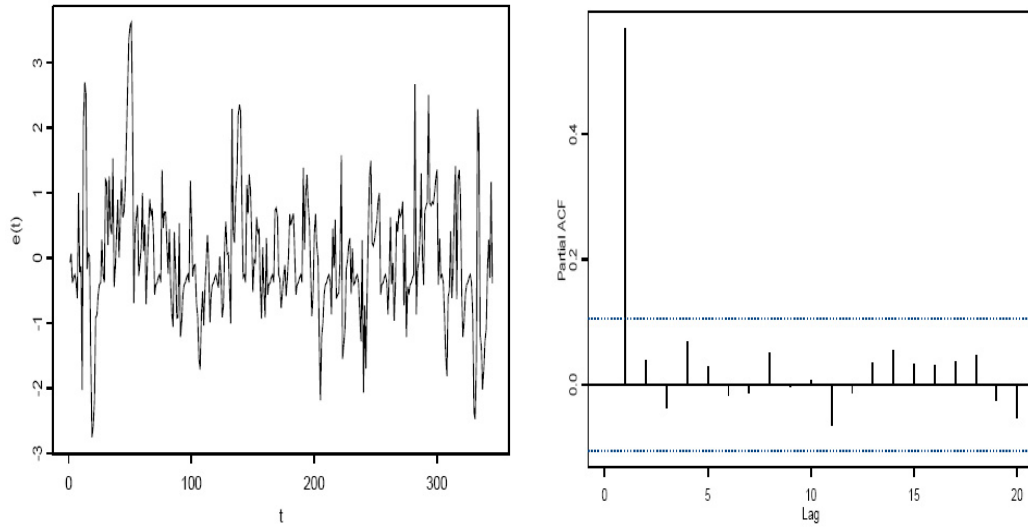


Figure 4.17: (a) Time series for  $e(t)$  (b) Partial ACF plot for  $e(t)$

the  $AR(p)$  model is written as

$$e(t) = a_1e(t-1) + \dots + a_pe(t-p) + n(t),$$

where  $n(t)$  is the model residual.

As for the load at AP 472,  $p$  is selected to be 1 and the fitted  $AR(1)$  model is

$$e(t) = 0.5689e(t-1) + n(t) \quad (4.1)$$

with the residuals  $n(t)$  being normally distributed with mean 0 and variance 0.6349.

**Normalized ARIMA based time-series forecasting** Using the normalized ARIMA model denoted by equation (4.1), we can predict the traffic load during the next hour, corresponding to time  $(t+1)$ ,  $X_{472}(t+1)$ . First, a point prediction for  $e(t+1)$  can be obtained as

$$\hat{e}(t+1) = 0.5689e(t);$$

then  $Y(t + 1)$  can be predicted as

$$\hat{Y}(t + 1) = \mu_{h(t+1)} + \sigma_{h(t+1)} \times \hat{e}(t + 1).$$

Finally, a point forecast for  $X_{472}(t + 1)$  is obtained by back-transforming  $\hat{Y}(t + 1)$ ,

$$\hat{X}(t + 1) = \hat{Y}^4(t + 1).$$

. Also, we can define the  $\epsilon$ -tolerance prediction intervals for this point prediction algorithm.

Note that the normalized ARIMA algorithm uses the model in (4.1) and actual traffic to perform forecasting. A different version of this algorithm is the normalized ARIMA multi-step ahead one (NAMSA).

### **Normalized ARIMA multi-step ahead time-series forecasting (NAMSA)**

Using the same three-week data (as in the other prediction algorithms), this normalized ARIMA multi-step ahead time series forecasting performs as follows. As Figures 4.18 and 4.19 illustrate, the prediction error ratio of the AP 472 (hotspot id 18) has a mean, median, and SD of 1.42, 0.72, and 3.77, respectively. Its correct percentages are 17.5%, 9.17%, and 6.67%, for a 25%, 10%, and 5%- tolerance prediction interval, respectively. The corresponding percentages for P1 are 20%, 10% and 6.67%, and for P2 20%, 18.33%, 16.67%, respectively. For a 25%-tolerance prediction interval, P3 with a two-hour window size and  $(a, b, c)=(1, 0, 0)$  has a 24.17% correct prediction percentage.

Figures 4.18 and 4.19 also illustrate the mean and median prediction error ratio of P3 with weights  $(a,b,c)= (1,0,0)$ , P3 with weights fitted using multiple linear regression, and NAMSA forecasting algorithm for all hotspots. Compared to the simple prediction algorithms P1, P2, and P3, the NAMSA algorithm results in better values for the mean and the SD of the error ratio (Figure 4.18). On

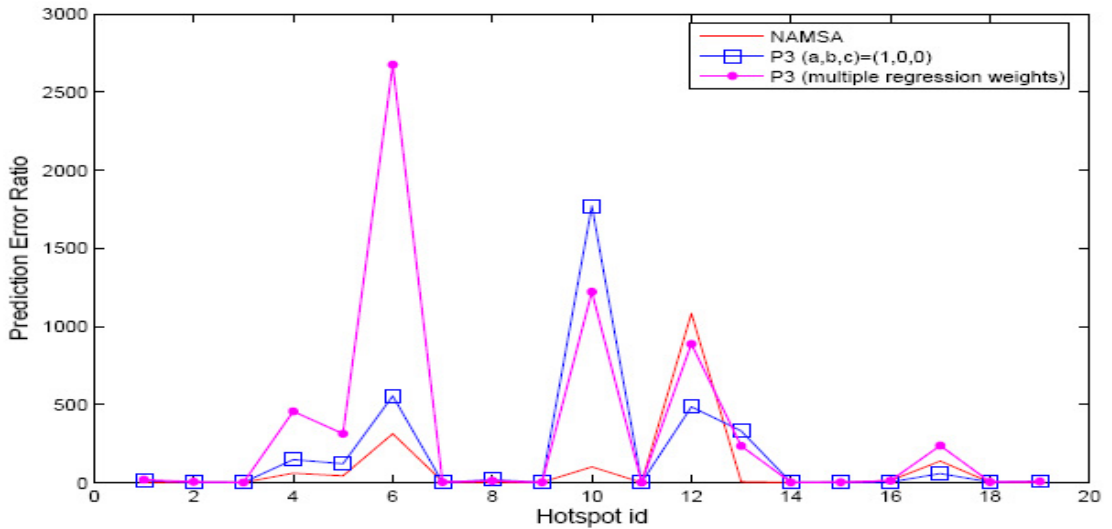


Figure 4.18: Mean prediction ratio for P3 and NAMSA forecasting algorithms for each hotspot

the other hand, the median of its error ratio is a bit worse than that of the P3 algorithm (Figure 4.19). This forecasting algorithm is a *multi-step-ahead* forecasting. That is, to predict a value, apart from the traffic model, the multi-step-ahead forecasting uses the recent *predicted* values instead of the actual ones. This makes the prediction even harder than the one-step ahead forecasting that uses the *actual* recent values like P3. We expect better performance when we use this algorithm for one-step ahead forecasting.

Note that P3 with weights fitted using multiple linear regression performs worse than P3 and NAMSA (with respect to both mean and median error ratio). This is due to the difference in the metrics used: The prediction error ratio (as defined in Section 4.5.1) is the ratio of the absolute difference of the predicted from the actual traffic over the actual traffic, whereas the multiple regression minimizes the square difference. When we use as metric the difference of the predicted from the actual traffic in square, we can observe that the mean of the

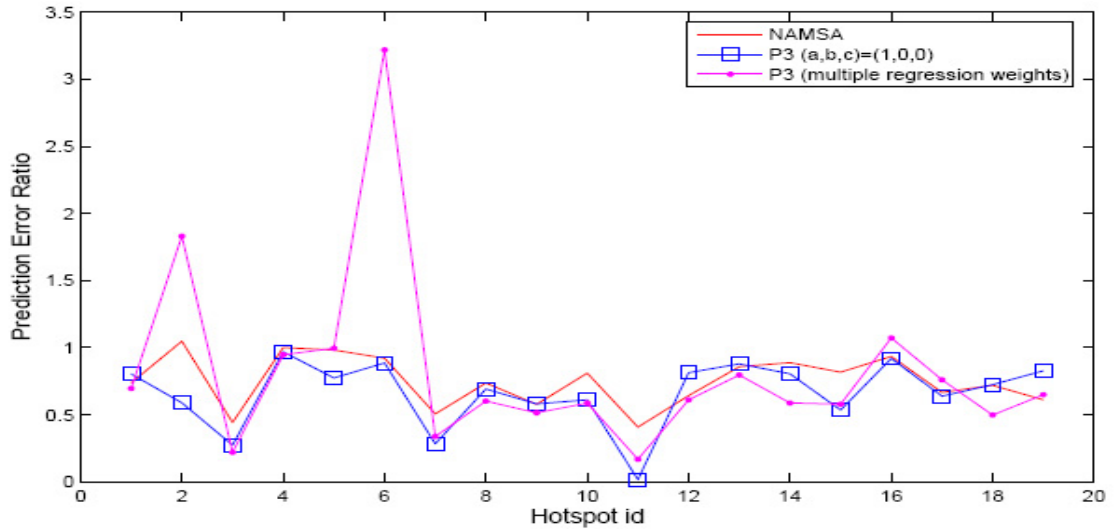


Figure 4.19: Median prediction ratio for P3 and NAMSAs forecasting algorithms for each hotspot

overall improvement of P3 (with multiple regression) for hotspots reaches 26%. Furthermore, we found that the dominant regressor in the weighted sum of P3 is history (for all hotspots). Specifically, in average, the recent history predictor participates in P3 with a percentage of 43.8% while historical mean hour and historical mean hour-of-day percentages are 41.1% and 15.1%, respectively.

## CHAPTER 5

### Multi-level application-based traffic characterization

In this section we characterize traffic of the UNC wireless infrastructure at the transport layer. Using packet header data, transport layer flows are reconstructed and categorized into application types using a graph-based method called *BLINC*. SNMP data are correlated with transport layer data allowing us to characterize the traffic of clients and APs as well.

The main goal of this study is to perform a multi-level application-based characterization of wireless traffic and explore how wireless channel dynamics and the inherent characteristics of a wireless network affect user behavior in terms of the application types used. We intend to explore which are the dominant and most popular application types throughout the network and identify application usage patterns at the infrastructure-wide-, client-, and AP-level. To achieve this application-based characterization of wireless traffic we classify flows to application types using a novel, graph-based, method avoiding the shortcomings of a payload- or port-based-classification.

Such a characterization of traffic can be beneficial in a broad range of wireless network related studies and problems. Recent studies [55] have observed a significant increase of P2P traffic accessed through a wide range of different P2P protocols, such as, the bit-torrent and direct connect. P2P systems consume a

large fraction of a network's available bandwidth since they are often used to exchange large files among users having thus a significant effect on overall network performance. Network administrators can make use of our classification to limit excess P2P traffic at certain locations of the network, such as, lecture halls or during peak hours of the day. Moreover, most modern tools-methods that classify transport layer flows to application types are able to detect flows that are due to malicious behaviors like Denial of Service Attacks (DoS), worms, or spamming. Such a classification can reveal crucial vulnerabilities of a network that may require administrator's attention. Characterizing traffic at the client and AP level is crucial to both building application usage profiles and designing better admission control and resource management mechanisms. Finally, real-time multimedia applications like VoIP have QoS requirements that are not always easy to guarantee in shared-medium networks, such as, a WLAN. Understanding usage trends is crucial to both designers of new protocols that provide QoS support and developers of new high-throughput and multimedia-friendly standards.

The main contributions of this section are the following: the most popular and dominant applications are Web and P2P accounting approximately for 81% of total network traffic. Most users are also dominated by these two applications. Although some applications like scanning activity and network management, do not transfer significant amounts of traffic, they are responsible for nearly 18% of the total flows in our trace. The vast majority of the clients appear to use the wireless network for one specific application that dominates their traffic. Moreover, while building-aggregated traffic application usage patterns appear similar, the application-cross section varies within APs of the same building. Comparing flow sizes between UNC's wired and wireless component we noticed that large file transfer flows such as Ftp and P2P are heavier in the wired network. We also noted that application usage profiles between more mobile and less mobile



clients are very similar in terms of the share of client’s traffic per application type. Finally, the AP-level characterization of wireless traffic revealed an interesting dichotomy among APs, in terms of their dominant application type and download and uploading behavior. To the best of our knowledge this is the first study that characterizes traffic of a campus-wide WLAN in terms of the application types used avoiding the known port limitation.

The datasets used in this section are packet header data collected from the egress router of the UNC campus-wide network along with SNMP polls and Syslog messages. For some of the tasks included in the current section we needed to cross correlate some of these datasets.

## 5.1 Data preprocessing

Performing an application based characterization of transport layer flows at the network, client, and AP level, requires first a significant amount of data preprocessing. This preprocessing includes:

- (a) reconstructing transport layer flows from packet header data;
- (b) classifying flows into application types;
- (c) adding MAC address, AP, and building info to flows;

### 5.1.1 Reconstructing transport layer flows from packet header data

As mentioned in section 3, packet header data were collected using a high precision DAG card from the egress router of UNC’s campus-wide network. Packet header data were stored both in “pcap” and “ASCII” format. We used CAIDA’s *CoralReef* suite [91] to process packet header data and reconstruct transport

layer flows. The algorithm employed by CoralReef suite to reassemble flows from packet headers uses the transport layer 4-tuple (source, destination IP address and source, destination), the protocol type, and a parameter indicating the flow timeout. CoralReef uses this parameter to expire active flows between hosts in the following way: it records all packets exchanged between hosts  $A$  and  $B$  (denoted by source and destination IP addresses found in the packet headers) through a specific pair of ports. It considers the flow to be active for up to “timeout” seconds after the last packet exchanged between hosts  $A$  and  $B$ . After the expiration of this timeout interval, it reassembles all packets recorded for that transport layer 4-tuple into a flow and reports a summary for that flow. All packets exchanged between the two hosts, through the same pair of ports, after the expiration of the timeout, are considered to be part of a new flow.

While tuning CoralReef to process a packet header trace, the timeout parameter must be selected carefully. Note that different application types use an application-layer timeout to expire connections after a period of inactivity. The timeout used while reassembling flows must be large enough to encapsulate the application layer timeout. HTTP by default terminates an idle connection after 60 seconds of inactivity while the corresponding value for Mail and Ftp is 300 seconds. Among other applications too, 300 seconds was the largest timeout used so we set the timeout parameter of CoralReef to 300 seconds.

### **5.1.2 Classifying flows to application types**

Classifying transport layer flows to application types accurately has been one of the main interests of network administrators and researchers of both wired and wireless networks.

### 5.1.2.1 Port and payload based classification

The classification of transport-layer flows based on port numbers may lead to significant amounts of misclassified traffic due to the following reasons: Many modern applications, such as, modern P2P protocols, audio and video streaming applications, and online games, use dynamically assigned port numbers. As a result, many different application types use overlapping port ranges making it very hard to distinguish amongst them by just looking at the port number. Apart from that, numerous applications, such as, malware or P2P may try to masquerade their traffic under well-known “non-suspicious” ports, such as, port 80. Indeed, there have been studies proving the inefficiency of port based classification of Internet traffic to application types [32, 55].

Another widely adopted method for classifying flows into application types uses a preamble of the payload. This method is based on the fact that most application types place a signature within the first 40 bytes approximately of the payload of each packet sent over the Internet through that application. Using RFCs, public documents, documentation and manuals of protocols, and reverse engineering, a list of signatures per application and per protocol is compiled. During processing packets of each flow the first, let’s say,  $n$  bytes of the payload are matched against the signatures of the list in order to identify the application type or protocol used.

Although this method can yield a very accurate classification it has several drawbacks that may discourage using it in certain cases. The first shortcoming of this method is that it requires keeping at least 40 bytes of the payload for each packet increasing the storage required for the trace. Especially for the case of high-speed backbone links of large LANs to the Internet, monitor points may fail to capture and store packets seen on a link at the same speed they are transmitted

resulting thus in a significant number of packet losses. Another inherent problem of the payload based classification is that prefixes used by applications may not always be known or even worse they may change from version to version. Finally, that are legal and privacy issues that must be considered before keeping payloads also apart from packet headers.

### 5.1.2.2 Classifying transport layer flows using BLINC

To avoid the shortcomings of payload-, and port-based classification, we employed the BLINC tool [31] which performs classification of flows into applications types based on the transport-layer footprint of the various application types.

**A. Background** BLINC was developed to classify transport layer flows to application types without taking into account either port number or preamble of any size from a packet’s payload. Instead of classifying individual flows to application types, BLINC associates Internet hosts with applications and then classifies their flows accordingly. In order to associate an Internet host to a specific application it analyzes the behavior of the host at three network levels:

- (a) social level;
- (b) network level;
- (c) application level;

This three-level analysis results in associating a host, and thus its flows, with one or more applications. Each one of the three-levels of analysis is intended to capture different aspects of a host’s behavior.

Analysis at the *social level* captures the popularity of a host in terms of the number of other Internet-side hosts with which it communicates. Intuitively,

this level focuses on the diversity of the interactions of a host in terms of its destination IPs and the existence of user communities. Analysis at the social level requires only access at the source and destination IP addresses.

The *functional level* captures the behavior of the host in terms of its functional role on the network, that is, whether it is a provider or consumer of a service, or whether it participates in collaborative communications. For example, hosts that use a single source port for the majority of their interactions are likely to be providers of a service offered on that port. Analysis at the functional level makes use of the source and destination IP addresses along with the source port.

Finally, analysis at the *application level* captures the transport layer interactions between hosts with the intent to identify the application of origin. At the first place, it provides a classification using only the 4-tuple (IP addresses and ports), and then the classification is refined by developing heuristics that exploit additional flow information, such as, the number of packets or bytes transferred as well as the transport layer protocol. For each application, BLINC captures host behavior by empirically derived patterns called *graphlets*. Graphlets represent the interaction between hosts and the corresponding ports that they use through graphs. The core idea of BLINC is that hosts using different applications will be represented through different graphs. Having a library of these graphlets BLINC tries to associate a hosts behavior with a specific graphlet and thus with a specific application (figure 5.1).

However, even with the use of graphlets, there may be cases ,such as, complex or similar graphlets, that may need disambiguation or further refinement. Along this direction, BLINC is augmented with a set of heuristics that are applied in the aforementioned cases.

*Heuristic 1. Using the transport layer protocol* The protocol information can

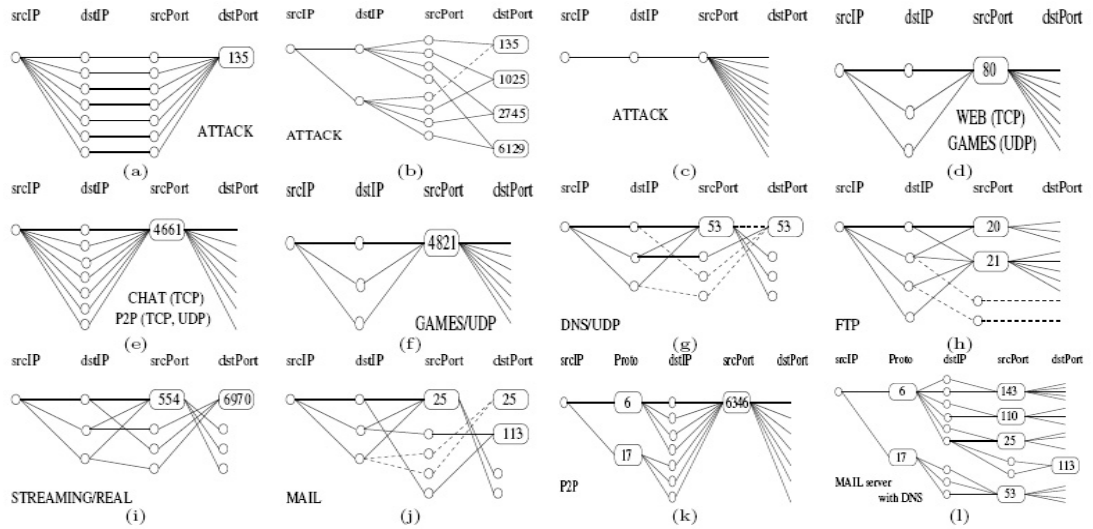


Figure 5.1: Graphlets: visual representation of transport-layer interactions for various applications (originally developed in [31])

be used to distinguish similar graphlets into three groups:

- (a) applications that use TCP: *Web, Ftp, Chat, Mail, and P2P*;
- (b) applications that use UDP: *Network Management, Online games*;
- (c) applications that may use both protocols: *P2P, Streaming*;

For example, while graphlets for Mail and Streaming appear similar, Mail interactions are performed only over TCP.

*Heuristic 2. The cardinality of sets* Number of ports vs. number of IP addresses is used to discriminate similar graphlets that are characterized by different behaviors, such as, Web and P2P or Network Management and online gaming.

*Heuristic 3. Per flow average packet size* Many applications have similar behaviors in terms of packet sizes. For instance, the majority of online gaming, malware, and spam-assassin applications initialize flows whose packets are of

constant size. Focus however is not on the actual size of the packet but rather on the fact that packets have the same size across all flows of the same application.

*Heuristic 4. Community heuristic* This heuristic examines IP addresses that reside in specific domains to identify sets of hosts that provide various kinds of services at a specific port.

*Heuristic 5. Recursive detection* Hosts offering specific types of services can be recursively identified by the interactions among them (variation of the community heuristic). For example Mail or DNS servers communicate with other such servers and use the same service port both as source or destination port across different flows.

*Heuristic 6. Non payload flows* Non-payload or failed flows are usually an indication of attacks or of clients of a P2P network that try to connect to other peers that have gone off line.

As far as the performance of BLINC is concerned, it manages to classify successfully into application types 80-90% of total traffic of the trace being analyzed on average. The accuracy of the classification, considering the results of a payload based classification as the ground truth, is always more than 95%.

Using BLINC in the packet header trace collected for the WLAN of the UNC campus-wide network, we were able to classify into application types 86.7% of the flows. The application types detected where: *Web, Chat, P2P, Mail, Ftp, Streaming, Network Management, Online games, Spam-assassin*, and *Scanning activity* that is, *Port-scan* (used by Trojan and worms) and *Address-scan* (used to launch DoS attacks). Flows that could not be classified to any of the aforementioned categories were marked as *Unknown*.

### **5.1.2.3 Statistical based methods for classifying flows into application types**

Recent studies [30, 54] have applied statistical techniques to probabilistically assign flows to application types. These methods apply clustering techniques to group flows into a predetermined number of groups. Criteria used to perform this classification are:

- average flow duration;
- average packet size per flow;
- statistical properties of the in-flow packet inter-arrival times;

### **5.1.3 Correlating packet header data with wireless sessions**

In order to perform an application-based characterization of UNC WLAN's traffic, at the client, AP, and building level, MAC layer information is needed for each flow. This information may contain the APs that the flow has traversed and the MAC addresses that has initiated the flow. For the client level characterization, assuming that one IP address corresponds to one client may lead to misleading results since as a client roams across APs of the wireless infrastructures he switches VLANs and thus possibly associates to a new IP each time. Apart from that, if a client reassociates to the wireless infrastructure after a long period during which he was disconnected, the DHCP server of the network may assign a different IP address to the corresponding MAC address. Due to these reasons it is possible for a specific IP address to have been associated to two different MAC addresses.

In order to add MAC layer information to the transport-layer flows, SNMP data were used. Syslog messages that could also provide MAC layer information



can not be coupled with flow-level data since Syslog messages are collected at the MAC layer where IP address information is not available. On the contrary, SNMP protocol runs at the application layer capturing thus both MAC layer and TCP/IP information. Based on client SNMP polls, wireless *sessions* were constructed. As figure 5.2 shows, a wireless session can be viewed as an episode in the interaction of a client and the wireless infrastructure: a wireless client arrives at the network, associates to one or more APs for some period of time, initiates transport-layer flows, and then leaves the infrastructure. During a wireless session, clients may initiate numerous flows through the corresponding APs to which they associate. For each wireless session a summary is created which includes the MAC address of the client, the IP address assigned to the corresponding MAC address, and the list of the APs to which the client has associated. Knowing the building to which each AP is located we can also have building level information for each flow. The timestamp of the first and the last SNMP polls that constitute a session are also used to derive the duration of the session.

In order to correlate a wireless session with a transport-layer flow, we needed to find all possible ways in which they might correlate. The possible ways in which a flow may correlate with a wireless session are the following:

- (a) the starting time of the flow falls within the duration of the wireless session;
- (b) the end time of the flow falls within the duration of the wireless session;
- (c) the start and end time of the wireless session fall within the duration of the flow;

Note however that client SNMP data provide accurate start times for associations, but end times are affected by sampling. For a polling interval of 5 minutes, the real end time of an association must be between the last polling time  $t$  and  $t$

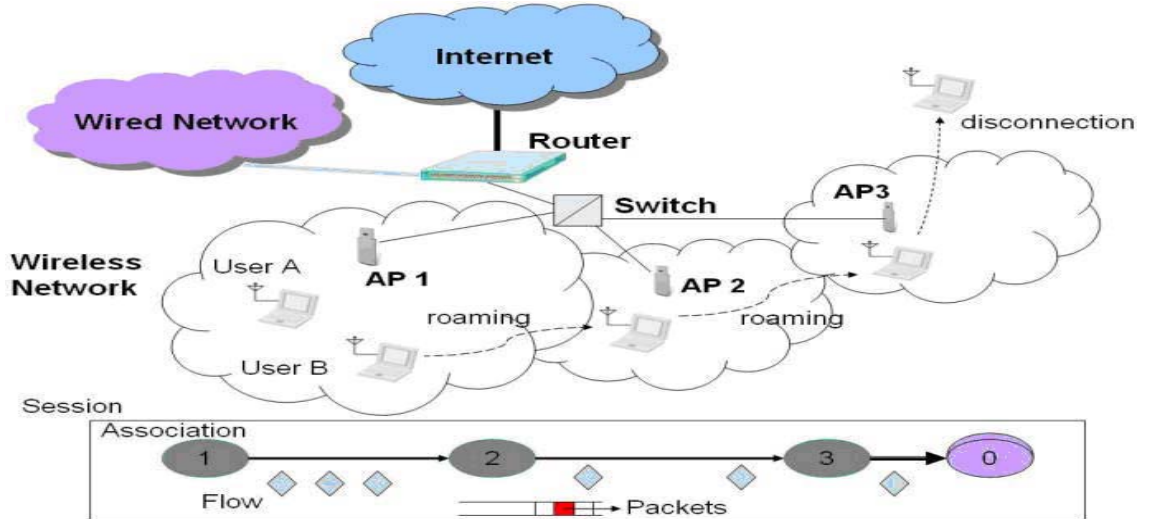


Figure 5.2: Wireless sessions and flows

+ 5 minutes. If a flow starts after  $t$ , the previous three rules will not be able to match the flow and the AP. This motivates a heuristic modification of rules (a) and (b), where an expanded association lifetime is used. This expanded lifetime is  $[\text{end}+5\text{minutes}, \text{start}]$ .

Applying the aforementioned rules along with the modification proposed we were able to correlate with wireless sessions 74.26% of the transport-layer flows which correspond to 77.6% of total network traffic.

## 5.2 Aggregate traffic characterization

While there have been several studies looking at the application cross-section at wired networks (e.g [3, 53, 60, 30, 54]), such attempts are limited in the case of wireless networks and are based on most cases on a port-based classification of flows [70, 52].

We first performed an application-based characterization of the total wireless traffic of the UNC campus-wide WLAN. This characterization includes identifying the dominant and most popular application types throughout the network, analyzing how flow sizes vary across different application types and comparing the application traffic mix of the WLAN being studied with other wired and wireless infrastructures.

### **5.2.1 Identifying dominant and most popular application types**

To identify the dominant application types throughout the network, we computed for each application type the percentage of: (a) total network traffic and (b) total number of flows that were accessed through it. We believe that the characteristics of a wireless network, such as, its limited resources and possible areas with no or weak coverage, would have a significant effect on wireless users' behavior. For example, due to large delays experience in a wireless channel, limited usage of delay sensitive applications, such as, streaming is expected. We also expect that due to limited throughput available at a wireless network, P2P should not have a significant share of total traffic.

As table 5.1 shows, the dominant application types throughout the network are Web and P2P accounting approximately for 81% of total traffic and and 80% of total flows. Specifically, P2P appears to be responsible for nearly 25% of total network traffic. Note however, that the way we collected our traces may underestimate the share of P2P traffic since it fails to capture all traffic that is exchanged within the WLAN. This table also shows that although some application types access insignificant amounts of total traffic, they contribute with a large number of flows. For example, although the Network managements along with scanning activity (address-, and port-scanning), are both responsible for 0.42% of total

bytes transferred, they account collectively for approximately 18% of total network flows. The effect of such applications on overall network performance is also significant since APs keep state for each flow that is initiated through them and periodically update their status tables consuming thus significant amounts of their resources. What is interesting in this table is that Mail is responsible for a rather small percentage of total number of network flows. However this is the effect of our monitor point placed after the egress router of UNC campus failing to capture any mail for which both the source and the destination reside inside the UNC WLAN. Finally, this table shows that other application types, such as, Ftp or Chat, transfer insignificant amounts of traffic. Note, once again, that the reason for which Ftp receives such a low share of total traffic is that our monitoring point fails to capture any Ftp requests to local (with respect to UNC WLAN) repositories.

Although byte and flow statistics reveal the dominant applications, in terms of network traffic, they only indirectly hint on the popularity of each application. We address *popularity* of an application through the number of clients that had at least one flow of that specific application. Note that by the term number of distinct clients we mean the number of distinct MAC addresses for which at least one flow was recorded through an application type. As Figure 5.3 shows, Web and Chat appear to be the most popular application types among clients. All of the clients have at least one flow through Web while that vast majority of them had used at least once a Chat application. An interesting observation that reveals significant security vulnerabilities of UNC's WLAN is that that vast majority of the clients have been scanned at least once by an address-scan or port-scan malware application. Figure 5.3 also shows that nearly 7 out of 10 wireless clients have used at least once a P2P application confirming results of recent studies that have shown an increased interest in P2P systems.

Application type	Flows (%)	Bytes (%)	Packets (%)
Network Management	9.95	0.42	1.54
Chat	2.05	0.48	1.47
Web	35.6	57.59	46.80
P2P	30.04	24.85	34.46
Online Games	1.11	0.01	0.075
Ftp	0.91	1.57	1.72
Mail	0.07	0.33	0.21
Address Sscan	6.40	0.12	0.58
Port Scan	0.39	0.32	0.28
Streaming	0.10	0.177	0.196
Unknown	13.2	14.09	12.64

Table 5.1: Percentage of total number of flows, packets and total traffic per application type

### 5.2.2 Distribution of flow sizes across application types

Exploring how the distribution of flow sizes varies across different applications can provide useful information about how certain applications perform over the wireless channel and also reveal user preferences. Due to the inherent resource limitations that characterize a wireless network, we would expect to notice very few large P2P and Ftp flows in terms of total traffic transferred. Moreover, for P2P users, frequent disconnections from the infrastructure will result in disconnecting from current peers and thus will further trigger peer discovery mechanisms. As a result, a large number of small-size P2P flows carrying control data will be observed. Concerning Web flow sizes, a large number of Web flows of the same size may be an indication of an on-going Denial of Service attack to one or more Web servers residing either in the Internet or inside the UNC WLAN.

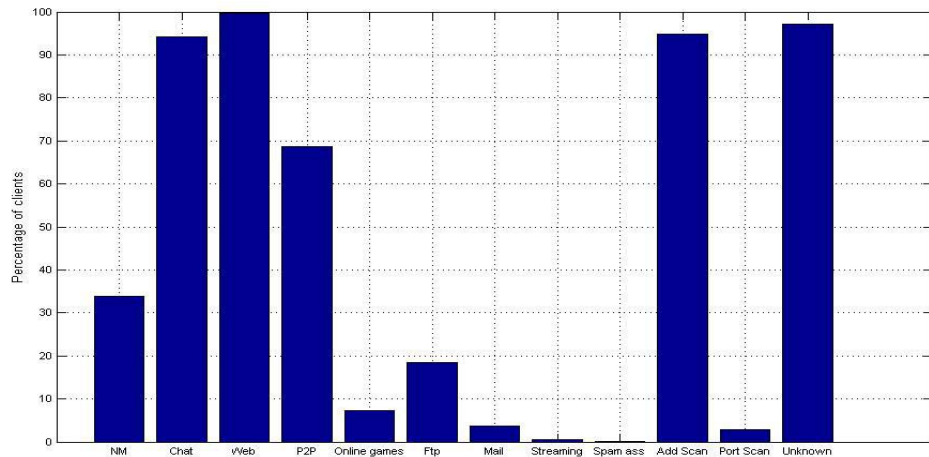


Figure 5.3: Popularity of application types

Figure 5.4 is a CCDF plot of flow sizes per application type. As this figure shows, Web flows appear to be heavier in terms of sizes than other application types with streaming flows coming next. It is also surprising to note that the distribution of Mail flow sizes appears to be so heavy. Note that 10% of the mail flows are larger than 10KBytes. This behavior can be attribute however to mails exchanged between hosts residing in the UNC WLAN and hosts that reside in the Internet that contain large attachments. Note that very few mail flows are larger than some MBytes which is the limit set for attachments for various mail servers. Looking at the distribution of flow sizes for P2P systems we note that nearly 80% of the P2P flows transfer up to 500bytes. P2P flow sizes transferring very few data are mostly control flows that either setup a direct connection with other peers or try to discover neighboring peers to connect to. This observation verifies our original intuition that we should notice a significant number of P2P control flows that would be the result of P2P users disconnecting from the wireless infrastructure. This figure also verifies our previous result that application types,

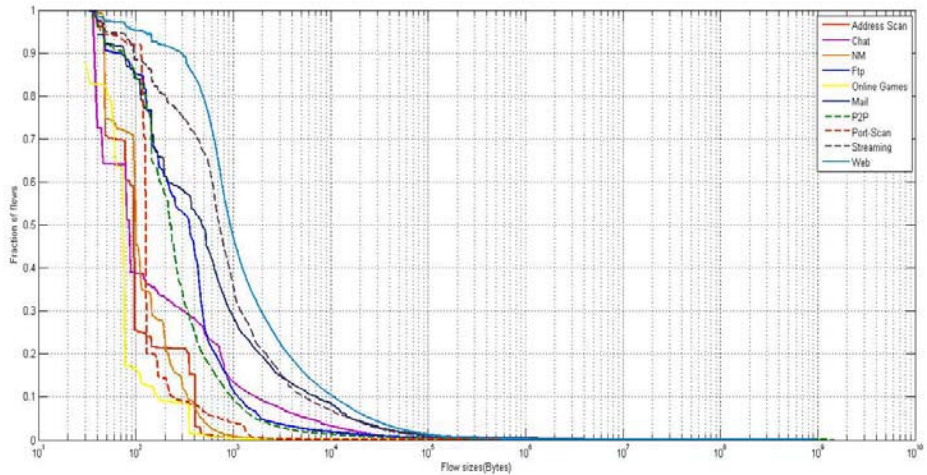


Figure 5.4: Distribution of flow sizes per application type

such as, scanning activity, Chat, and network management are not among the dominant application types. As it shows, flow sizes for Chat, address-scanning, port-scanning, and network management, are smaller than 1Kbyte for nearly 80% of the flows.

A common characteristic for flow sizes of all application types is that a significant portion of the flows transfer up to several KBytes while only a minority of them appears to initiate transfers as large as several Mbytes. This observation is very similar to the “mice” and “elephant” flows phenomena that appear in Internet backbone traffic [69]. To further explore if flows of specific applications exhibit similar phenomena, we calculated for the dominant and most popular application types, the percentage of flows that are responsible for 50% and 90% of total traffic through that application. For Web, 0.05% of the flows are responsible for nearly 50% of total Web traffic while 6.2% of the flows are responsible for 90% of total Web traffic. As far as P2P is concerned the corresponding percentages for 50% and 90% of total traffic are: 0.005% and 0.2% of total P2P flows. The

	BLINC wired	UNC Wired	UNC Wireless	Dartmouth Wireless
Web	37.5%	48.68%	57.59%	28.6%
P2P	31.9%	34.85%	24.85%	19.3%

Table 5.2: Percentage of Web and P2P traffic (bytes) across four different networks

same patterns are observed for the majority of other application types indicating that mice and elephants phenomena are very intense in our traces.

### 5.2.3 Comparative study with other wired and wireless infrastructures

At the final part of the application-based characterization of aggregate traffic, we contrast our findings with three other application-based characterization studies in wireless and wired networks, namely, two wired campus networks (BLINC, and UNC) and two wireless campus networks (UNC, and Dartmouth). Note that although a direct comparison is not straightforward due to the differences in the monitored networks, time of collection as well as the varying definition of application classes across studies, we can still observe general application trends.

To this end, we first compared the traffic share of the most dominant and popular applications (i.e., Web and P2P) of the UNC wireless network with the share of the same applications captured at the wired component of the UNC network (i.e., traffic originating from wired clients) within the same time interval. Similarly, we contrast our findings to the BLINC campus trace studied in the original BLINC work [31], and the findings from the Dartmouth wireless network [52]. Table 5.2 summarizes the percentages of Web and P2P traffic for each one of these networks. Note that the BLINC campus trace, and the two UNC traces were all classified by BLINC, hence the findings can be directly comparable; port



numbers were used in the case of the Dartmouth trace.

The results are remarkably similar for the two “wired” traces (BLINC and UNC campus networks), especially in the P2P case. On the contrary, the share of P2P traffic is significantly lower in both wireless traces amounting to approximately one fifth for Dartmouth and one fourth for the UNC of the total traffic. The most significant difference across the wireless traces is the share of Web traffic which is significantly higher in our traces. While such a difference may simply reflect different usage patterns across the two wireless networks, we speculate that the port based classification of Web traffic may have missed all traffic that was not destined to one of the well known ports for Web.

Finally, to further explore if application usage patterns are different between wired and wireless networks, we compared flow sizes for several application types over the UNC wired and wireless LANs. Figure 5.5 presents the CCDF plot of flow sizes between UNC wired and wireless LAN for Web, P2P, Ftp, network management, and address-scanning.

As expected wired flows are heavier than wireless ones for most application types. The most obvious reason for this is the larger throughput available at the wired network. This difference becomes more profound for the throughput-demanding applications, such as, P2P and Ftp and might imply that wireless users, being aware of the scarce resources of a wireless network, may choose to defer large file transfers either through Ftp or a P2P protocol until they get plugged to the wired network. We can also see that application types that perform standard operations (address-scanning and Network management) and thus initiate flows of a predetermined size perform the same over the wired and wireless channel. Finally we can see that the distribution of Web flow sizes is very similar between wireless and wired network which is reasonable since Web browsing is

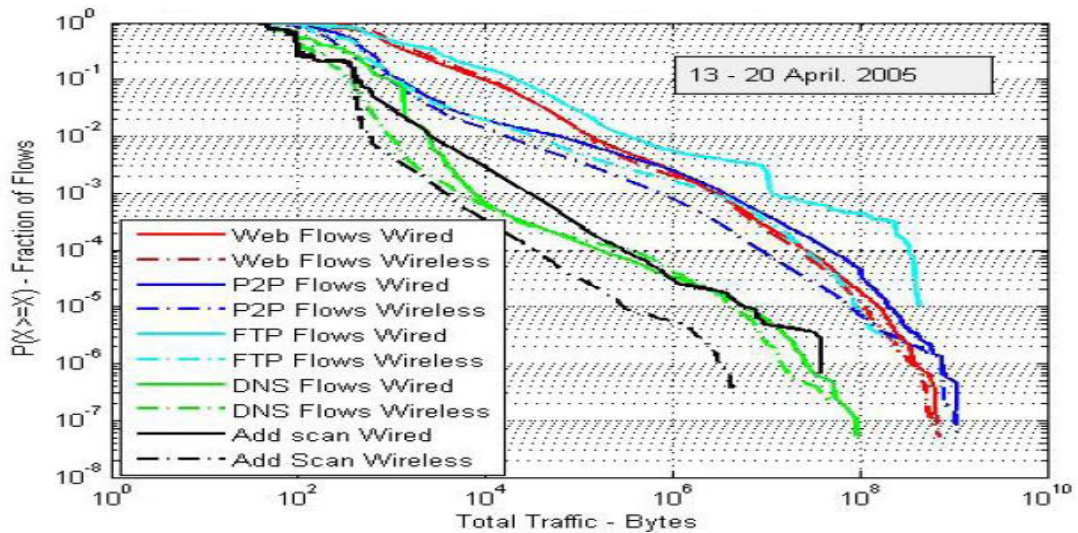


Figure 5.5: CCDF of flow sizes per application type. Wired vs. wireless LAN

neither delay nor throughput demanding.

### 5.3 Client traffic characterization

This section extends our aggregate traffic characterization to the client level. This section examines user behavior in terms of applications in order to gain a better understanding of the underlying application usage trends. Characterizing the client behavior is essential in designing more efficient AP admission control and selection mechanisms based on user profiles.

#### 5.3.1 Home application type per client

We define the *home application* of a client as the application that is responsible for more than  $x\%$  of that client's traffic. Wireless clients have strong application preferences, both in terms of number of flows and bytes. For example, for  $x$  equal

$x$	NM	Chat	Web	P2P	Games	Ftp	Mail	Strm	Ascan	Pscan	Unkn
<b>50</b>	0.40%	1%	88.8%	2.36%	0%	0.13%	0.21%	0.1%	0.03%	0.03%	5.64%
<b>75</b>	0.13%	0.77%	78.16%	1.5%	0%	0.07%	0.16%	0.06%	0.01%	0.0115%	2.5%
<b>90</b>	0.06%	0.56%	59.26%	0.8%	0%	0.06%	0.06%	0.015%	0%	0%	1.1%

Table 5.3: Percentage of clients per home application

to 90%, nearly half of the wireless clients have a home application that is approximately half of the clients transfer nearly all of their traffic through a specific application type. Table 5.3 indicates the percentage of wireless clients that have a specific application type as a home application with various thresholds. The most prominent home application is Web, while P2P appears to be the home application for only a minority of the clients. It is also interesting how the traffic mix varies for clients without a home application. Even in this case most clients are still dominated by Web. The second largest share of their data is accessed either through P2P or an undefined application.

### 5.3.2 Effect of wireless network on client application usage patterns

While user preferences with respect to applications over the wireless network appear to have similar trends as to wired networks [89, 51] (i.e., Web and P2P dominate), it is unclear whether client behavior is affected by the application performance over the wireless channel. To shed some light on client behavior over the wireless network, we compare the characteristics of the dominant applications over the wired and the wireless networks.

As observed in our earlier study [29], the distribution of flow sizes for wireless users is very similar for both the wired and wireless component when looking at the aggregate traffic. Uploading and downloading flows, also follow very simi-

lar distributions in both the wired and wireless component. Flow sizes however appear quite different between wired and wireless when looking at specific application types. Looking back at figure 5.5 we notice that flow sizes appear smaller for bulk file-transfer applications, such as, Ftp and P2P over the wireless network. In the case of Web traffic the two curves look very similar with wired Web flows being slightly heavier than wireless ones, P2P and Ftp flows however, appear “lighter” in size in the wireless network, especially for larger flow sizes.

There are two potential explanations for this observation, one is application-dependent while the second is user-driven. First, especially in the case of P2P applications, packet losses or disconnected TCP flows severely affect performance; broken TCP connections will result in disconnecting from existing peers, which will further trigger peer discovery mechanisms and increase queue waiting times, resulting in decreased flow sizes. As we observed in an earlier study [29], the large number of retransmissions at the 802.11 MAC layer, increases both the packet delay and number of retransmitted or failed packets at the transport layer. This is consistent with figure 5.6 which compares the number of flows per client between Web and P2P in the wireless network. Note that while overall the Web flows are heavier in terms of bytes and number of transferred bytes per client, the number of flows per client is larger in P2P than in Web. Essentially, each P2P client appears to have a large overhead in the wireless network with numerous small flows, corresponding to control traffic. Secondly, users may avoid transferring large files over the wireless network because of the limited throughput indicated also in Section 5.1 that compares the P2P traffic in wireless and wired. This observation holds for both Ftp and P2P transfers.

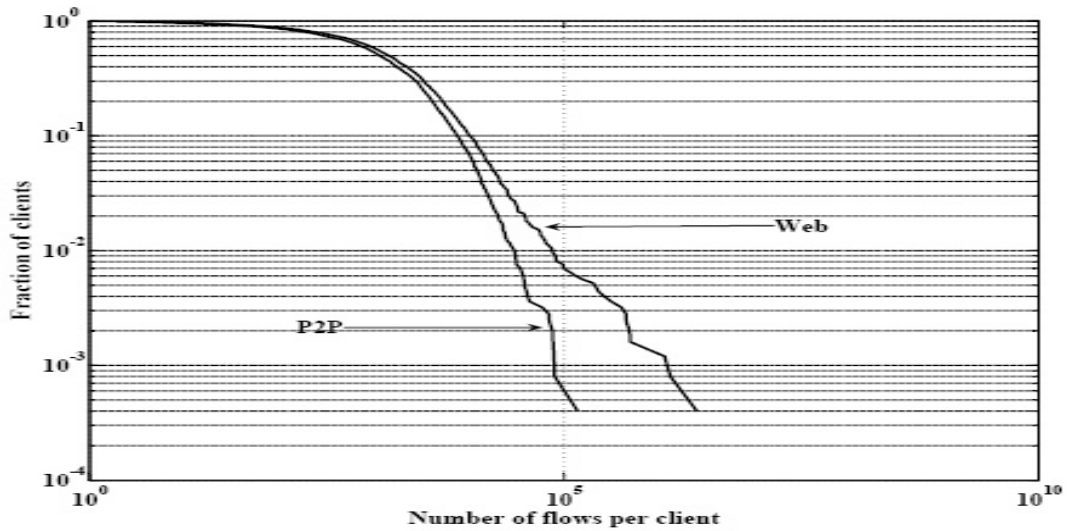


Figure 5.6: CCDF of number of flows per client for P2P and Web

### 5.3.3 Distribution of client traffic per application

Identifying the home application for each client may reveal application user preferences however it only indirectly addresses issues of how traffic is distributed among clients. Are there clients that are most active than other? are there heavy P2P or heavy Web users? Moreover, in section 5.2.2 we noticed that for most application types, a small percentage of total flows are responsible for the largest portion of traffic accessed through that application often referred to as “elephant flows”. Are these “elephant flows” flows uniformly distributed across clients or they are initiated through a specific subgroup of them? To answer these questions, we calculated for each client and each application, the percentage of total traffic of that application that was transferred by that client. Figure 5.7 presents the corresponding percentages of total application traffic per client for the dominant and most popular application types.

As this plot shows, there are very few clients that are responsible for more

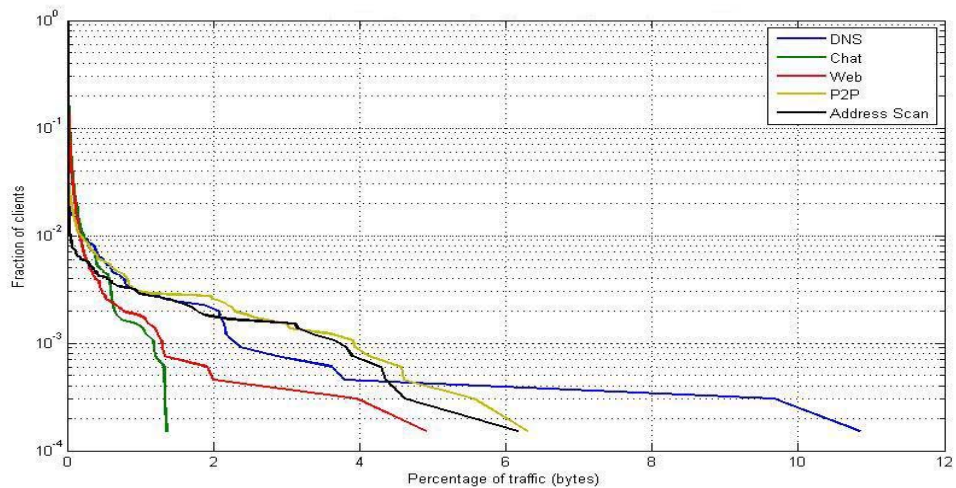


Figure 5.7: CCDF of percentage of total traffic per application type for each client

than 1% of the total traffic of a specific application. For example for Web, less than 0.001% of the clients are each one responsible for more than 10% of total Web traffic observed in the wireless component of the UNC network. This plot also reveals that there are heavy P2P users in our network being responsible for approximately more than 4% of total P2P traffic each. It is also interesting to note that the majority of address-scan traffic is transferred through a rather small fraction of total wireless users probably suggesting a set of compromised hosts that are used as “zombies” to launch a DoS attack. Finally, in order to see if “elephant flows” for the dominant applications (Web, P2P) are transferred through a specific set of clients, we calculated for these application types, the percentage of clients that are responsible for  $x\%$  of their total traffic.

As table 5.4 shows, 1.5% of the UNC’s wireless users are responsible for approximately half of total Web traffic observed in the traces while 24% of them

$x\%$ of total traffic	50%	70%	90%
Web	1.56%	6.29%	24.29%
P2P	0.18%	0.33%	0.98%

Table 5.4: Percentage of clients that transfer  $x\%$  of total network’s Web and P2P traffic

are responsible for the vast majority of Web traffic. However the phenomenon of some clients transferring nearly all of the traffic of a specific application type, is significantly more intense in the P2P case where nearly 1% of the clients transfer 90% of total P2P traffic suggesting that “elephant” P2P flows are initiated through a specific set of clients while for the case of Web the are dispersed to a larger number of clients.

#### 5.3.4 Effect of mobility on application usage patterns

The last part of our client traffic characterization focuses on exploring how mobility affects client application usage patterns. Our intuition is that application usage profiles between stationary and mobile clients should have different patterns. First of all, very mobile clients who frequently disassociate from the infrastructure and then re-associate back or go off-line should have more transient behaviors, such as, Web browsing, Mail and Chat. For example, frequent disassociations from the infrastructure should discourage mobile clients from initiating large file transfers through Ftp or P2P. This means that for certain applications we expect that flow sizes will be heavier for stationary users.

Although there have been several studies on wireless networks that have proposed schemes to address mobility related issues [50, 75, 27, 28, 68], we rely on a simple method to distinguish between mobile and stationary clients since our main focus is on identifying application usage preferences between mobile

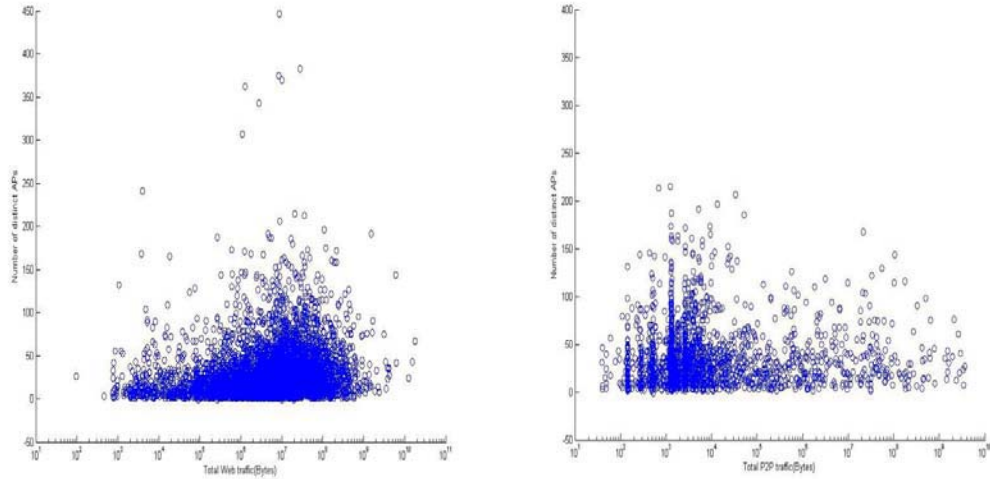


Figure 5.8: Total traffic vs. number of distinct APs visited for (a) Web (b) P2P

and stationary clients and not on inferring a realistic mobility model. The metrics used to express a wireless client’s mobility are the number of distinct APs that a client has visited and the number of visits of that client at all APs of the infrastructure (the notion of *visit* was first introduced in Section 4.3.1).

Figure 5.8 is a scatter plot of total Web and P2P traffic for all clients vs. the number of distinct APs visited by each client. As this figure shows, as the number of APs that a client has visited increases, total traffic that is accessed through either Web or P2P is not affected. Note that the majority of both heavy and light Web users visit up to 50 APs while heaviest P2P users also associate to at most 75 distinct APs. To stress-test our hypothesis we focused on the most mobile and on the least mobile clients of the infrastructure. We define two classes of clients based on the number of distinct APs that they visit: *Top mobile*: Top 5% of the clients in terms of number of distinct APs visited and *top stationary*: Bottom 5% of the clients in terms of number of distinct APs visited.



$x$	NM	Chat	Web	P2P	Strm	Ascan	Pscan	Unkn
<b>Top mobile</b>	4.34%	2.97%	76.91%	1.61%	0.07%	0.086%	0.2%	13.23%
<b>Top stationary</b>	1.40%	3.46%	78.38%	0.76%	0.18%	0.75%	0.04%	14.92%

Table 5.5: Mean percentage of a client’s traffic through each application type. Most mobile vs. most stationary clients

For each one of the two classes, the mean percentage of a client’s traffic that was accessed through each application type was computed.

As table 5.5 shows, top mobile clients access on average 1.61% of their traffic through P2P while the corresponding percentage for stationary clients is 0.76%. For all other application types, top mobile and top stationary clients access very similar fractions of their traffic through each application type which shows that apart from P2P, application usage profiles are similar between mobile and stationary clients.

## 5.4 Application usage patterns across APs and buildings

In this section we extend our aggregate and client traffic characterization at the AP level. Such a characterization can provide valuable information to administrators who wish to perform resource provisioning and guarantee QoS for certain application types at certain locations of the network. For example, APs that experience large delays and access a significant share of their traffic through Web or Streaming may require different settings, such as, using IEEE802.11e which provides QoS support. Moreover, administrators may wish to either filter out or limit P2P usage at certain APs that reside in over-crowded areas. Finally, as security has become a crucial issue for most wired and wireless networks, ad-

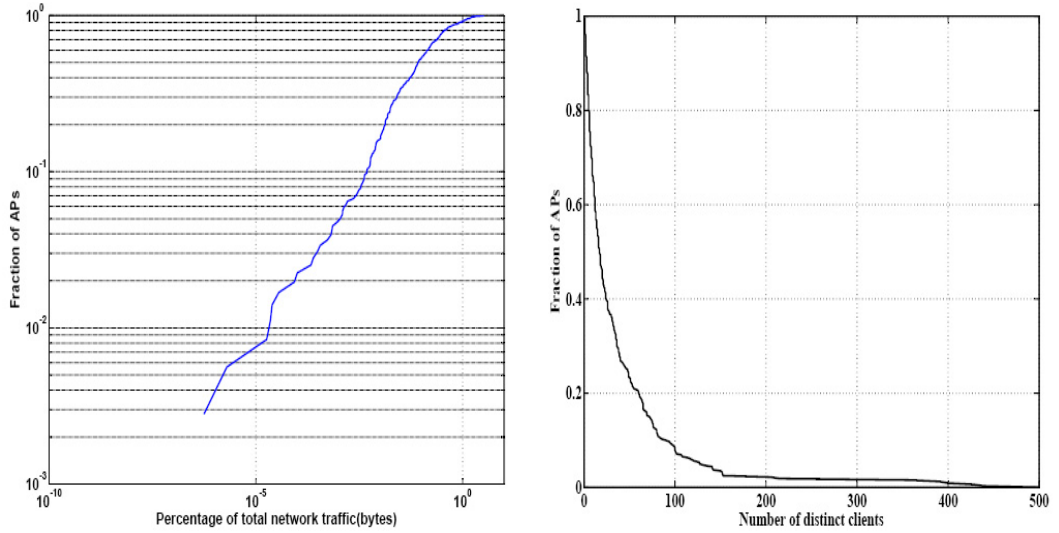


Figure 5.9: (a) CDF of percentage of total network traffic across APs. (b) CCDF of the number of distinct clients across APs.

ministrators need to have information about both locations of the network where numerous attacks are noticed and of misbehaving hosts.

As observed in previous studies, the overall distribution of traffic across APs is not uniform. Few APs are responsible for the largest amount of traffic. The reason for such a skewed distribution of traffic among APs is the varying popularity among APs. To validate this assumption, we plot the CCDF of the number of clients across APs in Figure 5.9(a). This figure reveals that few APs are significantly more popular than others having been visited by more than 400 clients. As expected, these APs correspond to the ones with the highest traffic aggregation.

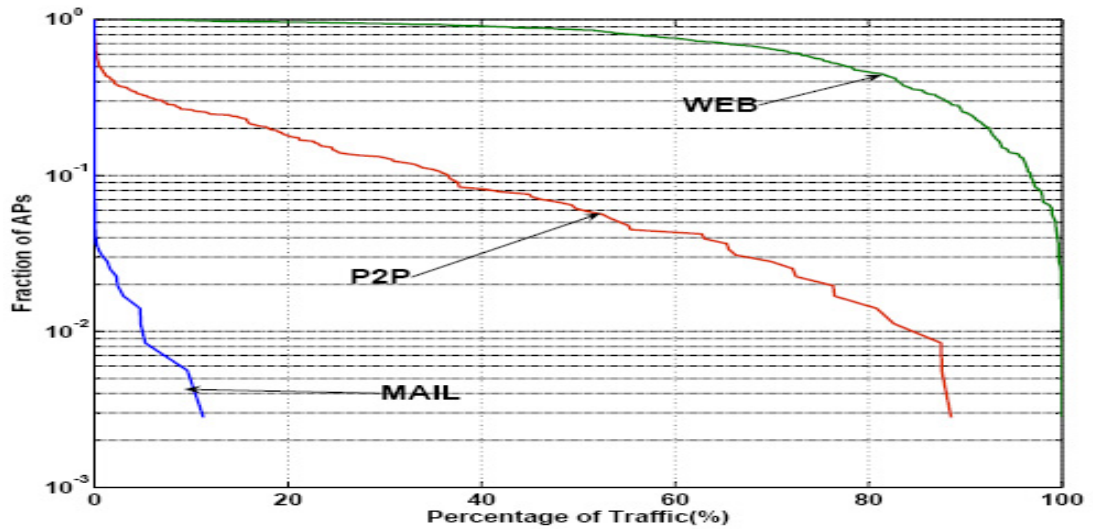


Figure 5.10: CCDF of the percentage of each APs traffic per application type

#### 5.4.1 Application usage patterns across APs

We first examined how the distribution of traffic varies across APs. In order to examine if the share of traffic of each application is similar across APs or whether specific applications dominate particular APs we computed for each AP the percentage of its total traffic that was access through each application type. Figure 5.10 presents the corresponding CCDF for Web, P2P, and Mail.

As this figure shows, the distribution of traffic across APs varies with the application type. While high percentages of Web traffic appear in most of the APs, a small portion of them, roughly 10%, access a significant amount of P2P data. It is also interesting to explore how the traffic mix varies across APs that are not dominated by Web. Figure 5.11 shows the percentage of an AP's traffic that is accessed through other application types for APs which access at most 50% of their traffic through Web Figure 5.12 shows the combined percentage of Web and P2P traffic across APs sorted by their Web traffic percentage for APs

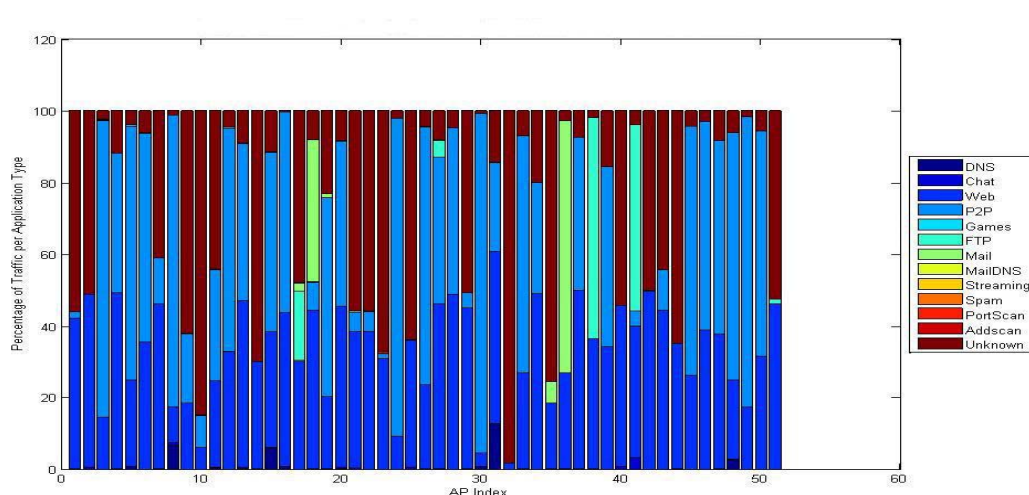


Figure 5.11: Percentage of AP traffic through other applications for APs with less than 50% of their traffic through Web

which access at most 75% of their traffic through Web.

These two figures reveal that APs that are not dominated by Web form two groups: the first group contains APs that are dominated by P2P while the second group contains APs that are not dominated by P2P or Web, however Web along with P2P are responsible for the largest portion of this AP's traffic. Although Web is the most popular and dominant application type at the network level, this is not the case at the AP level. Enough of the APs are not dominated by Web and access significant percentages of traffic through other application types. Figure 5.11 also shows that other application types rarely dominate any of the APs with the exception of three APs being dominated by mail traffic and one by Ftp.

#### 5.4.2 Home application type per AP

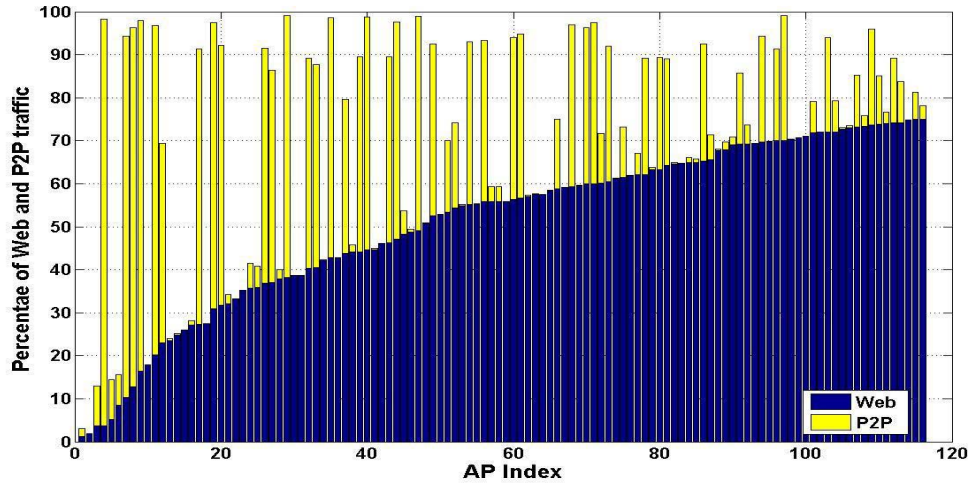


Figure 5.12: Combined share of traffic for P2P and Web for APs with less than 75% of their traffic through Web

$p$	Web (%)	P2P (%)	Ftp (%)	Mail (%)	Unknownn (%)
50	85.9	6.17	0.28	0	4.2
75	55.8	0	0	0	0.84
90	25.52	0.28	0	0	0

Table 5.6: Percentage of APs with a home application.

To study in more detail what application types dominate the traffic of APs we say that an AP has a certain type of application as *home application*, if more than  $p\%$  of its total traffic is of that application type. Table 5.6 presents the breakdown of APs that have such a home application when half of the traffic of the AP belongs to a specific application type. Approximately 4% of the APs are dominated by unknown traffic, while for roughly another 4%, a home application could not be defined since no application type is dominating this AP's traffic. Overall, there is an application preference towards specific APs in the wireless

Building Type	APs	Web (%)	P2P(%)	Ftp(%)	Mail(%)	Unkn(%)
Academic	165	79	6	-	1.8	3
Administrative	36	66	8.3	-	-	8
Clinical	16	62	6	-	-	6
Athletic	15	53	20	-	-	6
Residential	42	83	5	-	-	2
Business	15	73	-	5	-	-
Library	12	86	-	-	-	-
Conference	9	55	-	-	-	11
Theater	4	90	5	-	-	-

Table 5.7: APs per building category and the percentage of APs with a home application.

network. This is an important observation since it can direct traffic engineering decisions, such as, load balancing or filtering P2P traffic at certain locations of the network.

### 5.4.3 Application usage patterns across buildings and building types

To further examine the spatial variation of the application cross-section, we grouped APs based on their building category. Working at building level circumvents several problems emerging when working at AP-level: non amenability to statistical processing, higher sensitivity of monitored traffic variables to the short-term propagation conditions, and lack of scalability [2]. These categories reflect buildings with similar functionalities and allow us to examine whether the share of the application depends on these functionalities. We would expect for example to notice insignificant amounts of online games and P2P traffic at APs residing in academic buildings or near lecture halls.

AP ID	44	45	46	47	48	49	50	51	52
Web (%)	89	81	87	85	78	<b>61</b>	95	<b>45</b>	74
P2P (%)	0.01	0.09	0.1	0.1	0.2	<b>35</b>	0	<b>51</b>	<b>11</b>

Table 5.8: Web vs. P2P traffic share across APs of building ID 22.

To this end, a similar analysis is performed using the notion of the home application for an AP as defined in the previous section. Table 5.7 presents the number of APs for 9 building categories, and the percentage of APs for which a home application existed. There is a weak correlation between the building category and the number of APs that have a home application (e.g., Mail exists only in the academic buildings as a home application, while Ftp is present only in the business category). This reinforces our intuition that distinct APs may require different configuration settings depending on the application or the type of building functionality.

The uneven traffic distribution in the application cross-section that was observed across buildings exists also across APs of the same building. Table 5.8 presents the percentage of Web and P2P traffic for all APs located in building 22. This building was chosen randomly among the buildings with the largest number of APs. While in most cases Web traffic dominates the overall traffic share, there are distinct APs (highlighted in the table) in which P2P contributes with the largest amount of traffic. Note that these are not transient traffic phenomena, since our tracing period corresponds to several days.

#### 5.4.4 Download to upload asymmetry

Previous studies [26] have observed that certain APs are dominated by uploaders. In this section, we first explore if this phenomenon persists over time. Then, we

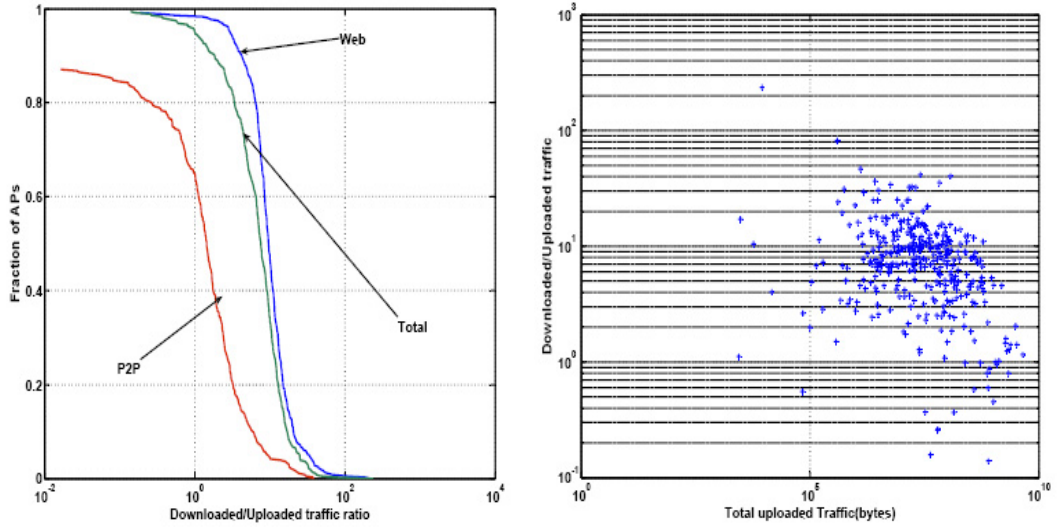


Figure 5.13: (a) CCDF of the asymmetry index for Web, P2P, and total traffic (b) Asymmetry index vs. uploaded traffic

examine which application is responsible for the asymmetry observed between the downloaded and the uploaded traffic. For that, we define the asymmetry index of an AP to be the fraction of the total downloaded to the total uploaded traffic (bytes).

Approximately 5% of the APs show asymmetry indexes less than 1 which means that more bytes are uploaded than downloaded by users associated to these APs. Figure 5.13 (b) is a scatter plot presenting the asymmetry ratio for each AP versus the total uploaded bytes at the same AP. It is interesting to note that especially APs with significant amounts of uploaded traffic show asymmetry indexes less or very close to 1, which implies symmetric usage of these APs. The asymmetry trend observed in this study using packet header traces of 2005 is very similar to the one observed in [26] using packet header traces of 2004. This suggests that the phenomenon of asymmetry between total downloaded and total uploaded traffic at APs persists over time. However in the traces of 2005



there were slightly fewer APs with asymmetry indexes less than 1 suggesting that overall downloading activity has increased.

To examine if such an asymmetry is application specific we focused on the asymmetry index per application. Intuitively, applications that are characterized by symmetric usage (e.g., P2P) could explain ratios less than one in specific APs; for example, heavy uploaders that are mostly associated to a specific AP could significantly affect the usage of that AP. For roughly 40% of the APs, the asymmetry index for P2P is less than 1, while this is true for only 1.4% of the APs when it comes to Web. As expected, uploading behavior is more intense in P2P than in Web, which highlights the need for application profiling of the various APs: network operators should dimension APs according to their application usage characteristics to exploit such phenomena. In addition, there was significant uploading behavior for chat and streaming applications although P2P is the main reason for almost all small asymmetry indexes observed in Figure 5.13 (b).

## 5.5 Application based characterization of wireless sessions

In the next chapter, extending our transport layer characterization, we are going to focus on evaluating models that describe how sessions arrive at a campus-wide wireless network and also how flows arrive within a session. We will also explore if these models can be used to characterize flow and session arrival rates for flows of specific application types. The question that we are interested to answer is whether sessions are homogeneous in terms of total traffic transferred per application. Are sessions dominated by a specific application type or there are more than one applications that contribute with a significant amount of traffic to these sessions?

$x$	NM	Chat	Web	P2P	Games	Ftp	Mail	Strm	Ascan	Pscan	Unkn
<b>50%</b>	0.32%	1.62%	88.64%	2.27%	0%	0.27%	0.19%	0.09%	0.039%	0.0072%	5.56%
<b>75%</b>	0.18%	0.98%	79.97%	1.57%	0%	0.22%	0.17%	0.08%	0.01%	0.0072%	2.88%
<b>90%</b>	0.079%	0.95%	65.42%	1.04%	0%	0.21%	0.13%	0.07%	0.03%	0%	0.63%

Table 5.9: Percentage of sessions that transfer  $x\%$  of their traffic through each application

For that reason, we calculated for each application, the percentage of sessions that access more than  $x\%$  of their traffic through that application type. Table 5.9 summarizes the corresponding percentages for  $x=(50\%, 75\%, 90\%)$ .

As this table shows, the vast majority of the sessions transfer more than 50% of their traffic through Web. What is interesting is that although we have seen that there are some APs that are dominated by P2P, there are very few sessions that access a significant portion of their traffic through P2P, suggesting that P2P is distributed among the various sessions. However, when  $x=90\%$ , less than 67% of the sessions are dominated by a specific application type with the vast majority of them being dominated by Web.

## CHAPTER 6

### Evaluation of parametric models for wireless traffic demand

In this section we extend our transport-layer characterization of a campus-wide WLAN's demand by suggesting and evaluating the performance of a set of parametric models that describe wireless traffic at the session- and flow-level (proposed in [11] published in the *Second annual international Wireless Internet Conference*).

Such models of wireless demand can prove valuable to performance analysis studies that wish to explore how widely deployed (or even new) protocols and mechanisms (e.g for load balancing or admission control) perform under realistic conditions. Since trace collection from a large campus-wide WLAN may not always be feasible or it may require a significant amount of time to collect and pre-process datasets, synthetic traces generated based on realistic models can consist input to simulation and testbed experiments.

The main goals of this study are twofold. At first we shall evaluate through simple statistical tests and through simulations the performance of a set of parametric models for wireless demand. Based on the proposed models and the parameters acquired during the first part of the analysis, we implement a synthetic trace generator (in a Matlab module) and contrasted synthetic traces with actual data.

One of the main contributions of this study is that we evaluate the performance (proving the goodness of fit) of the set of parametric models proposed in [11] through both statistical tests and simulations. Applying simple visual tests it is also shown that flow-level models can also be applied to Web flows apart from aggregate network flows. Finally we implement a synthetic trace generator based on the coherent parametric statistical models of the WLAN workload and show that the synthetic traces produced capture the trends observed in the actual data.

Both the modeling and evaluation methodology used at this section were based on transport layer connections extracted from packet header data of the UNC WLAN. For each connection we also have AP and application type information. Statistical tools that are used are: Grubb's test for detection of outliers, MLE to fit the parameters of models, random number generators, CCDF and QQ-plots

## 6.1 Modeling methodology

This section provides a brief summary of the modeling methodology used as proposed in [11]. The modeling methodology proposed in this work is hierarchical in that it is based on two fundamental components namely, the wireless session and transport layer connection (also referred to as "flow").

As said in section 5.1.3, a wireless session can be viewed as an episode in the interaction of a client and the wireless infrastructure: a wireless client arrives at the network, associates to one or more APs for some period of time, and then leaves the infrastructure. It was preferred over modeling individual association-disassociation sequences, whose dynamics in wireless LANs can change dramatically due to small changes in the network layout, physical environment, or

network/client equipment avoiding thus too network dependent characteristics. Models based on network dependent characteristics can not be used in studies that either explore these characteristics or propose mechanisms that shape them. For example, in the context of WLANs, modeling the precise sequence of associations and disassociations inside sessions is too network-specific, since small changes in the network layout, physical environment, or network/client equipment can dramatically change association/disassociation dynamics. Therefore, the simulation model should not impose a priori a certain sequence of associations and disassociations. This requirement is satisfied when sessions are the subject of modeling. The simulated session may end up having completely different association dynamics, but the corresponding workload is preserved.

In this hierarchical modeling approach, wireless sessions consist the higher level unit of wireless traffic including all transport layer connections that were initialized within these sessions by clients who have associated to one or more APs. On the other hand transport layer connections provide a finer level of modeling wireless traffic. Transport layer connections represent the interaction between a host of the UNC WLAN and a host residing on the Internet including all packets exchanged between these two hosts through a pair of ports. Working with transport layer connections is in line with the approaches followed in [49, 67, 86] and the principles of network-independent modeling from [87]. Simulating wireless workload based on this two-tier approach consists of simulating sessions and the flows started inside them, leaving packet-level and association dynamics to underlying mechanisms that are independent of our model.

The two-tier modeling approach developed in [11] relies on parametric models for the traffic demand variables. When compared with empirical models, they provide better insight to the properties and the dynamics of the modeled quanti-

ties. In parallel, they are more adequate in summarizing datasets and make their comparison straightforward. The parametric models proposed for both session- and flow-level traffic variables are statistical distributions. To derive the distributions that best fit the flow- and session-level variables, extensive use of formal and visual statistical tools was made.

## 6.2 Proposed models

The actual variables that are modeled are through a statistical distribution are:

1. session arrival times;
2. flow inter-arrival times within a session;
3. flow sizes;
4. number of flows per session;

Table 6.1 summarizes these variables, along with the proposed model for each one and its Probability Density Function.

Modeled variable	Model	Probability Density Function (PDF)
<b>Session arrival</b>	Time-varying Poisson( $\lambda(t)$ )	$N$ : # of sessions between $t_1$ and $t_2$ $\lambda = \int_{t_1}^{t_2} \lambda(t) dt$ , $Pr(N = n) = \frac{e^{-\lambda} \lambda^n}{n!}$ , $n = 0, 1, \dots$
<b>Flow interarrival/session</b>	Lognormal	$p(x) = \frac{1}{\sqrt{2\pi x\sigma}} \exp\left[-\frac{(\ln x - \mu)^2}{2\sigma^2}\right]$
<b>Flow number/session</b>	biPareto	$p(x) = k^\beta (1+c)^{\beta-\alpha} x^{-(\alpha+1)} (x+kc)^{\alpha-\beta-1}$
<b>Flow size</b>	biPareto	Same as above

Table 6.1: Summary of models for network-wide traffic demand variables

## 6.2.1 Background theory on biPareto distribution and on Time-varying Poisson Process

This subsection provides some general background information about two of the distributions that are employed in [11] to model flow- and session-level variables namely the biPareto distribution and the time-varying Poisson process (also referred to as inhomogeneous Poisson process).

### 6.2.1.1 biPareto Distribution

The biPareto distribution is specified by four parameters  $(a, b, c, k)$ , whose complementary cumulative distribution function (CCDF) is given by:  $\left(\frac{x}{k}\right)^{-\alpha} \left(\frac{x/k+1}{1+c}\right)^{a-\beta}$  where  $x \geq k$ .  $x \succ k$  is the minimum value of biPareto random variable, which is a scale parameter. The CCDF initially decays as a power law with exponent  $a \succ 0$ . The in the vicinity of a breakpoint  $kc$  (with  $c \succ 0$ ), the decay exponent gradually changes to  $\beta \succ 0$ .

Essentially, the biPareto distribution has two Pareto tails on both ends of the distribution. On a log-log plot, a CCDF of the form  $\chi^{-\alpha}$  (a Pareto tail) would appear as a straight line with slope  $-\alpha$ . Thus, the log-log plot of a biPareto CCDF has two nearly linear regimes, with slopes  $-\left(\frac{c}{1+c}\alpha + \frac{1}{1+c}\beta\right)$  and  $-\beta$ , respectively. This property of the distribution makes it a good choice for modeling the number of flows per session and flow sizes. Its parameters can be estimated via maximum likelihood estimates.

### 6.2.1.2 Time-varying Poisson process

Poisson process [92, 93] is one of the most important models used in queueing theory. It is often used to model the arrival process of certain events, such as,

customers at a queue. For the case of networks, a Poisson process is a viable model when the packets originate from a large population of **independent** users. Mathematically the process is described by the so called counter process  $N(t)$  which indicates the number of arrivals that have occurred in the interval  $(0, t)$ .

A Poisson process can be described in three equivalent ways: (a) Poisson process is a *birth* process: in an infinitesimal time interval  $dt$  there may occur only one arrival. This happens with the probability  $\lambda dt$  independent of arrivals outside the interval. (b) The number of arrivals  $N(t)$  in a finite interval of length  $t$  follows the Poisson( $\lambda t$ ) distribution with

$$P\{N(t)=n\} = \frac{(\lambda t)^n}{n!} e^{-\lambda t}$$

Moreover, the number of arrivals  $N(t_1, t_2)$  and  $N(t_3, t_4)$  in non-overlapping intervals  $(t_1 \preceq t_2 \preceq t_3 \preceq t_4)$  are independent.

(c) The interarrival times are independent and obey the Exp( $\lambda$ ) distribution:

$$P\{\text{interarrival time} \succ t\} = e^{-\lambda t}.$$

### 6.3 Evaluation of parametric models for wireless traffic

In this section we evaluate how the models described in 6.1 for the corresponding session- and flow-level variables fit with the actual data. The evaluation is performed through visual and simple statistical tests and also through simulations. For the simulations we make use of *Matab's* built-in random number generators for the distributions used in our models.



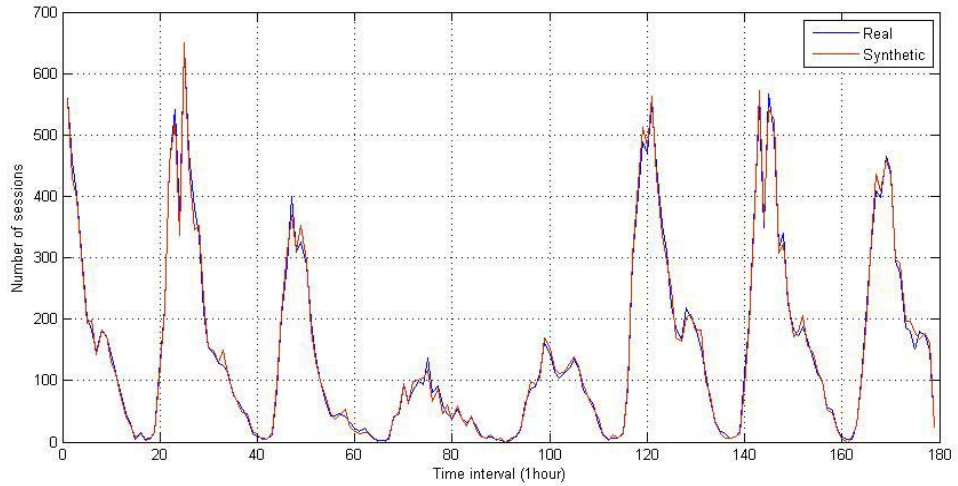


Figure 6.1: Number of sessions per hour. Empirical vs. synthesized data

### 6.3.1 Fitting sessions with a time-varying Poisson process (TVPP)

To evaluate the performance of a TVPP fit for the session inter-arrival times within a time block, we generated theoretical session interarrivals times and compare them against the real ones. We first extracted session inter arrival times within a block of length  $L=1$  hour from the real trace, assuming that the Poisson arrival rate is constant within this 1 hour block. We then simulated the time-varying Poisson process in the following way: for each simulated interval of length  $L$ , we consider that we have a Poisson process with a constant rate  $r$ . We fix this rate from the actual data. Then, for this interval we generate session interarrival times using the inverse CDF of the exponential distribution with rate  $\lambda = r$ . Session interarrival times are generated up to the point that their cumulative sum will become  $\succ L$ . Having generated the session interarrival times within each session, we extract session arrival times and construct the time series of these arrivals.

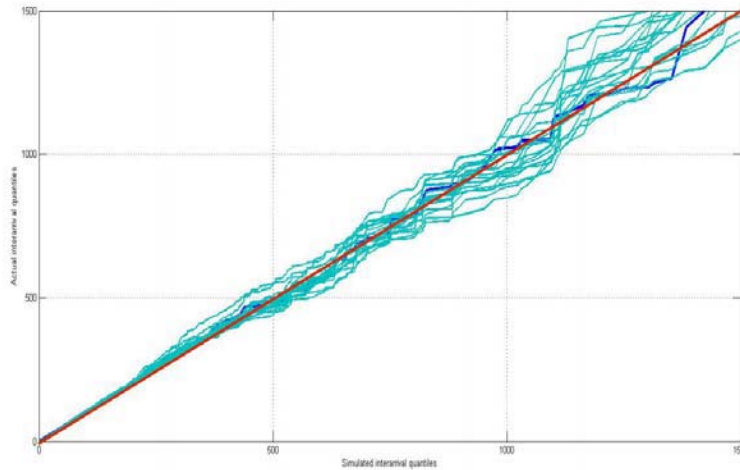


Figure 6.2: QQ-plot of simulated vs. empirical session interarrivals

Figure 6.1 shows the number of sessions observed per time interval for the 179 hours of the real data and the synthetic based on the simulated TVPP. As this plot shows, the deviation between the real and the synthetic data is insignificant for all hours of the trace implying a very accurate fit of session arrivals through a TVPP. To further check on the validity of the proposed model, we run 30 simulations of the time-varying Poisson process and plotted the simulated session inter-arrivals against the actual ones. Figure 6.2 presents the QQ-plot of the simulated session interarrival times against the actual ones. As this plot shows, the quantiles for the vast majority of the interarrival times fall well within the simulation envelope.

### 6.3.2 Fitting flow sizes through a Bipareto distribution

The statistical analysis carried in [11] reveals that flow sizes can be accurately described using a biPareto distribution. To evaluate the efficiency of this model

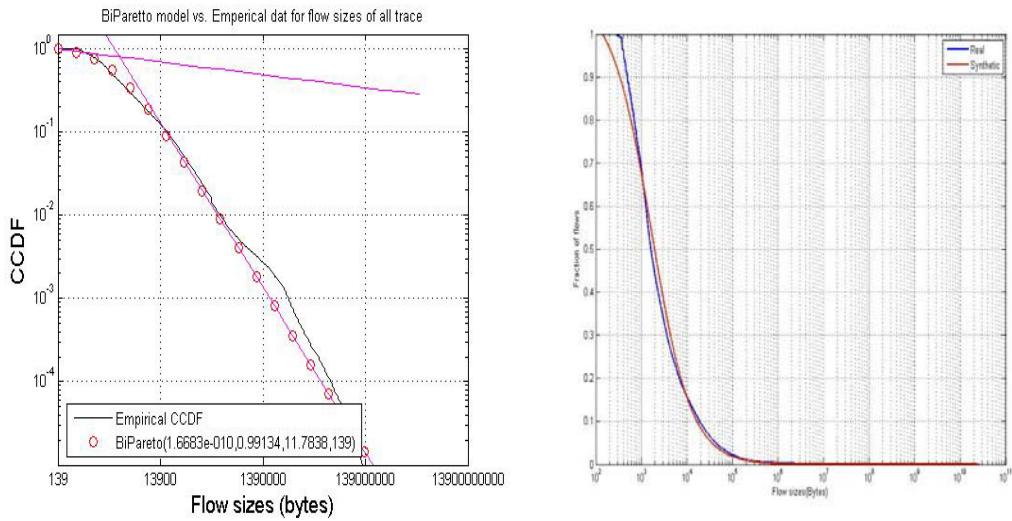


Figure 6.3: CCDF of (a) theoretical vs. empirical flow sizes (b) simulated vs. empirical flow sizes

flow sizes for the 11,188,727 flows that exist in our real trace are extracted. Using MLE over all flow sizes, the parameters of the Bipareto distribution are  $(a, b, c, k) = (1.6683e-010, 0.99134, 11.784, 139)$ .

Figure 6.3 (a) shows the joint CCDF of a Bipareto distribution with the parameters computed using MLE over the actual flow sizes. We see that the fit is excellent for most of the distribution with biPareto clearly capturing the transition in the slope between the body and the heavy tail of the empirical distribution. To further explore the performance of the biPareto model, we employed a custom biPareto random number generator to synthesize flow sizes using the parameters computed using MLE. Figure 6.3 (b) presents the corresponding CCDF of the synthesized flow sizes against the actual ones. For the vast majority of the flow sizes, the two distributions are almost identical with the exception of flows with rather small sizes where the theoretical model slightly overestimates the actual flow sizes.

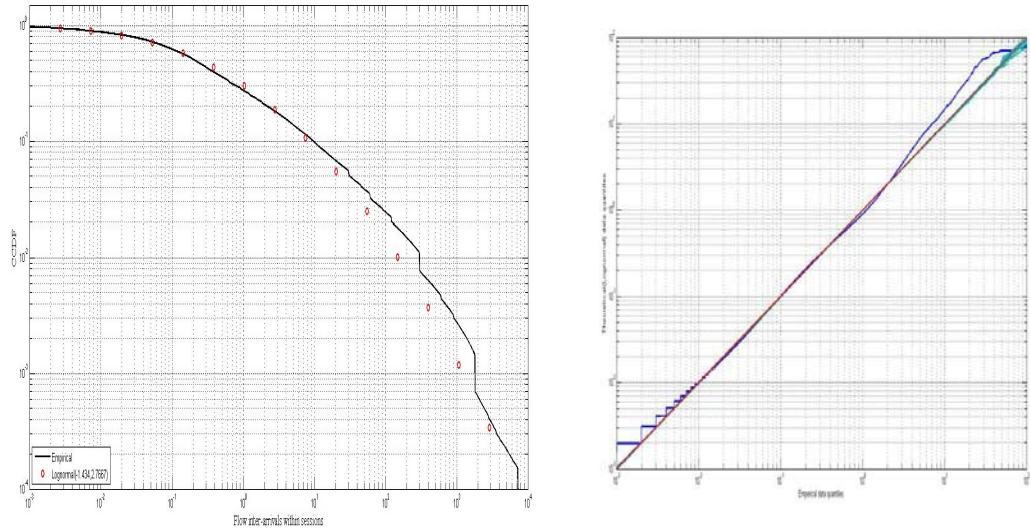


Figure 6.4: Flow interarrival times: (a) CCDF of theoretical vs. empirical (b) Lognormal QQ-plot

### 6.3.3 Lognormal fit for flow inter-arrivals within a session

The parametric model proposed in [11] to describe flow inter-arrival times within a session is the lognormal. We first extracted flow inter-arrival times within session from the real data and then used MLE to calculate the  $\mu$ ,  $\sigma$  parameters of the model. Figure 6.4 (a) is a joint CCDF of the theoretical flow interarrivals derived from our model with parameters  $\mu=-1.434$  and  $\sigma=2.7667$ , and the empirical ones. As this figure shows, our model follows the trend observed in the real data for 90% of the flows approximately while for the rest 10% of the flows it slightly underestimates flow inter-arrival times. While more complex models, *e.g* an ON/OFF model, may provide a better approximation, the proposed lognormal model certainly provides a reasonable description of the data using only two parameters.

The lognormal quantile plot for the empirical data is also shown in 6.4 (b).

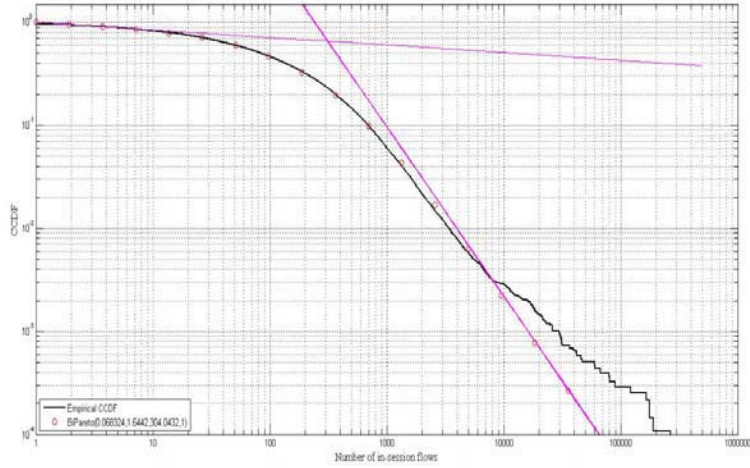


Figure 6.5: CCDF of number of flows per session. Theoretical vs. fitted

Due to the large number of flows in the trace, the QQ-plot was drawn for a randomly selected session among the ones with the largest number of sessions. The blue curve corresponding to the empirical quantiles follows the red diagonal line closely for all of the quantiles. The simulation envelope is very narrow in this case, and shows that there are some significant deviations from the lognormal model in the upper part.

### 6.3.4 Evaluating biPareto fit for number of flows per session

The last variable that is used in this two-tier modelling approach is the number of flows per session which is described through a biPareto distribution. Figure 6.6 plots the CCDF of the fitted distribution with parameters  $(a, b, c, k) = (0.068324, 1.6442, 304.04, 1)$  against the empirical data in a logarithmic scale.

The empirical distribution of the number of flows per session matches well the proposed biPareto model for sessions with up to approximately 10,000 flows.

The fit is worse at the tail due to some sessions that have a rather large number of flows. These sessions can be due to abnormal or outlying users behavior. Clients that were compromised by an address-scan attack may then be used as zombies and take part in a DoS attack by initiating a large number of small-size flows to other internet side hosts. Moreover, clients using a bit-torrent P2P application that experience frequent disconnections from the infrastructure, will initiate a large number of control flows to discover and reconnect to existing peers.

To explore however how significant is the effect of sessions with such outlying behavior we compare the number of flows per hour observed in the real trace with the number of flows generated by the biPareto model. In order to create the time-series of the number of flows per session, we need to have the flow arrival times which in turn require to know the session arrival times. Although section describes 6.5 how all these models can be used in conjunction to produce synthetic traces, a brief summary is presented here. The procedure of generating a synthetic time series of flow arrivals includes the following steps:

- (a) For each time interval of the simulated duration;
- (b) Session arrivals are extracted from session inter-arrival times obtained through a TVPP;
- (c) For each session, flow inter-arrival times are generated which are transformed to absolute flows arrival times;

We synthesized a trace of 179 hours triggering all aforementioned models with their corresponding parameters computed using MLE and drew a joint timeseries plot ( 6.6) for the number of flows per hour for the emperical and simulated data.

As this plot shows, there are specific time intervals during which the synthesized traces underestimates the number of flows in our trace. The most prominent deviation of the real from the synthetic trace is observed between the *100-th* and

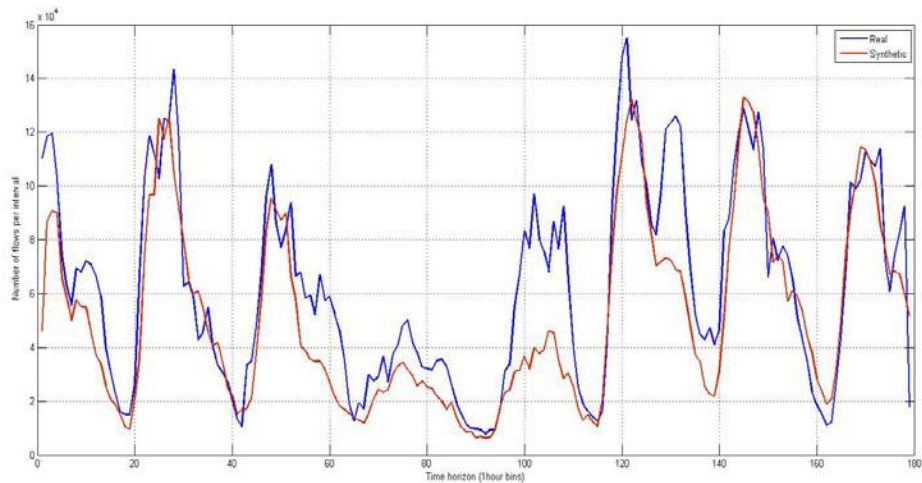


Figure 6.6: Time series of number of flows per hour. Synthetic vs. empirical trace

112-th hours approximately.

#### 6.3.4.1 Removing outlying sessions

To further improve the performance of the proposed models we employed the Grubb's test (Appendix C) to remove outlying sessions, i.e. sessions with an excessive number of flows. Applying Grubb's test on an iterative manner resulted in removing 0.2% of the total sessions which were responsible for 6.7% of the total traffic observed for UNC's WLAN in the packet header trace.

The parameters of all session- and flow-level models were recomputed for the corresponding distributions using MLE having excluded first all outlying sessions and all flows initialized within these sessions from the real trace (Figure 6.2).

Using the new parameters for the flow-level variables, we repeated the process described in 6.3.4. Figures 6.7 (a) and 6.7 (b) show the joint CCDF between synthesized and empirical data for both the number of flows per session and flow

Modeled variable	Model	Model Parameters
Session arrival	Time-varying	Fixed from
	Poisson( $\lambda(t)$ )	Real trace
Flow interarrival/session	Lognormal	$(\mu, \sigma) = (-1.3343, 2.9788)$
Flow number/session	biPareto	$(a, b, c, k) = (0.0702, 1.8565, 366.6385, 1)$
Flow size	biPareto	$(a, b, c, k) = (1.716e-12, 1.0847, 23.3648, 139)$

Table 6.2: System-wide parameters for flow, and session variables

sizes.

As we can see the fit for the flow sizes is well in line with the empirical data for nearly 99% of the flows. What is however more interesting is the improvement in the fit of the number of flows per session after having removed the outliers leading to a smoother tail for the corresponding distribution which is accurately described by our model. The improvement is even more profound if we consider the time series of number of flows per hour. As Figure 6.8 shows,

## 6.4 Extending flow and session related models to the application level

The previous section evaluated the performance of the proposed flow- and session-level models at the infrastructure level considering flows from all applications through all APs. This section explores how these models perform when applied for flows of specific applications. Specifically, since Web was found to be the dominant application in terms of total traffic transferred (Section 5.2.1), we are going to explore if the models proposed in Section 6.2 can be used for the corresponding variables for Web flows only. Note also that in Section 5.5 we have



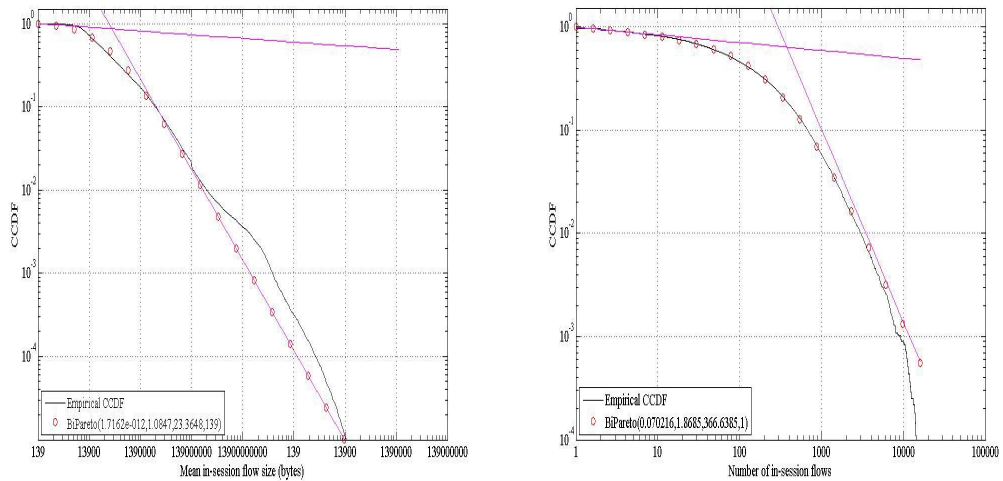


Figure 6.7: CCDF of synthesized vs. empirical data for (a) number of flows per session (b) flow sizes

observed that nearly 65% of the sessions transfer the vast majority of their traffic through Web.

#### 6.4.1 Evaluating a biPareto fit for Web flow sizes

Looking at the CCDF plot of Web flow sizes (figure 6.9) we can see that the distribution of Web flow sizes is a power-law like distribution with a very smooth tail (notice the log-log scales of the plot). Appropriate power-law-like distributions to fit such data are the exponential, pareto, biPareto, and Weibull distribution.

For each one of these distributions, we run MLE over the actual Web flow sizes to get the parameters that best fit the corresponding model. Based on these parameters and using Matlab's built-in random number generators for these distribution, synthesized flow sizes are generated. Note that we generate synthesized flow sizes for the same number of flows that appear in the real trace (Web flows).

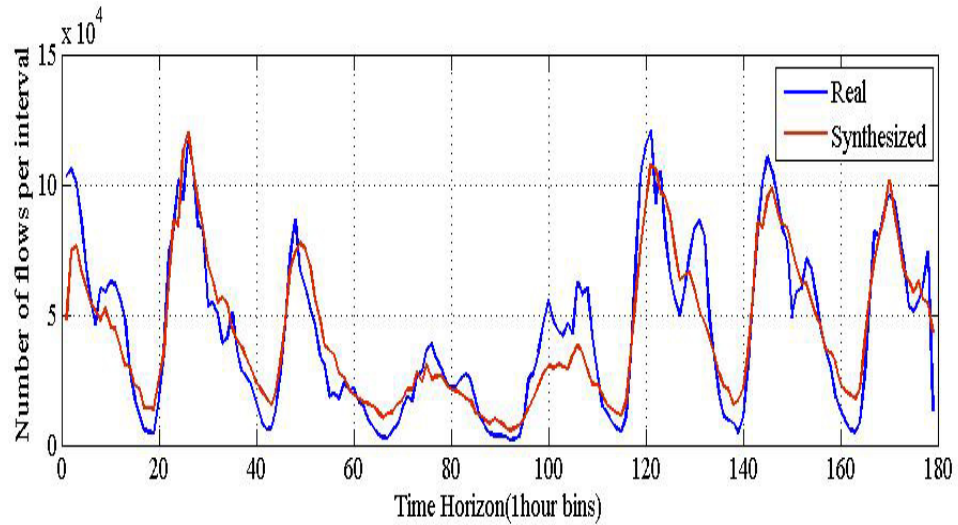


Figure 6.8: Timeseries of number of flows per interval. Emperical vs. synthetic

To evaluate which distribution provides the better fit, a joint CCDF of all synthesized flow sizes along with the actual ones is presented in figure 6.10. As it shows, for probability values ranging from 0.1 to 1, the biPareto model clearly provides the better fit for Web flow sizes. For flow sizes  $\succ 30$ Kbytes, the generalized Pareto distribution also provides an efficient fit, apart from the biPareto. Overall, the model that appears to be the most appropriate to describe Web flow sizes is the biPareto.

#### 6.4.2 Number of Web flows per session

When concerning all flows of the network, the model that fits accurately the number of flows per session is the biPareto. Figure 6.11 presents the CCDF plot of the theoretical against the actual Web flow sizes. As it shows, number of Web flows per session can be accurately described by a biPareto model with parameters  $(a, b, c, k) = (0.075, 2.1693, 425.019, 1)$ . The fit is very accurate for

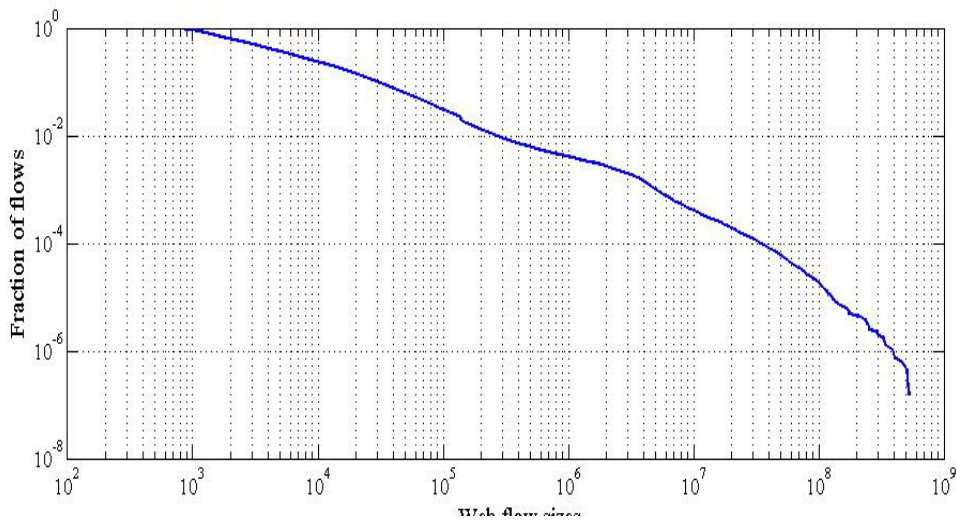


Figure 6.9: CCDF of Web flow sizes

all sessions clearly capturing the transition in the slope between the body and the heavy tail of the empirical distribution.

### 6.4.3 Evaluating a lognormal fit for the Web flow inter-arrival times within a session

In Section 6.3.3 we have noticed that a lognormal fit can provide an accurate approximation for the flow interarrival times within a session. To evaluate if this model can also be used to describe the interarrival times of Web flows within a session we extracted flow interarrival times for Web flows from the real trace and calculated the parameters of the lognormal model through MLE ( $\mu=-1.3627$ ,  $\sigma=3.033$ ).

Figure 6.12 (a) shows that the lognormal model performs very well for flow inter-arrival times of up to 10 seconds. For larger flow interarrival times, the proposed model slightly underestimates actual data but overall it provides a

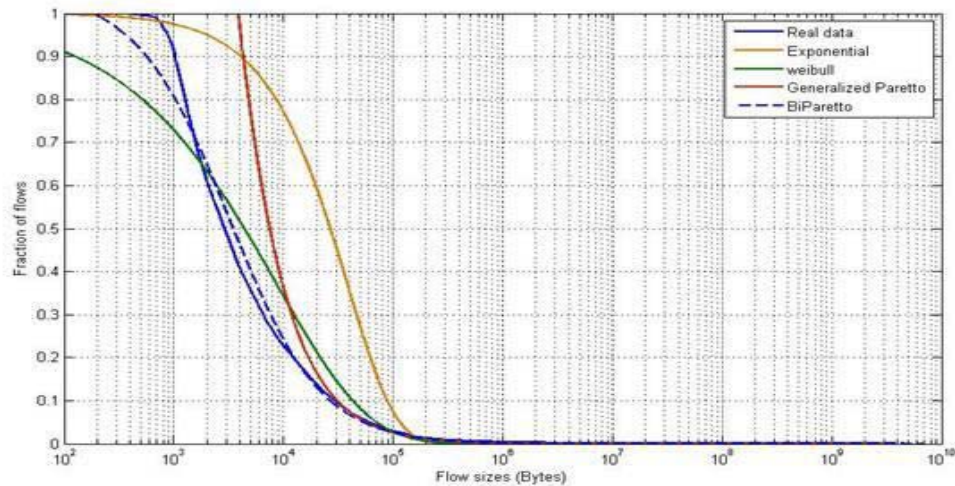


Figure 6.10: CCDF of Web flow sizes. Synthetic vs. empirical data

satisfactory fit using only two parameters as opposed to other more complex models.

## 6.5 Implementation of a synthetic trace generator

In the last part of this section we describe the implementation of a synthetic trace generator based on the coherent parametric statistical models for UNC's WLAN workload evaluated in the previous sections. This generator can be used for both generating synthetic traces and for validating proposed models. The output of the synthetic trace generator can be directly employed in simulation and testbed experimentation studies to generate more realistic user behaviors, while it allows for a better insight to the problem than empirical models. Network load can be simulated at both the client association and flow levels. The synthetic trace generator proposed is parametric in that the user can adjust the following parameters:

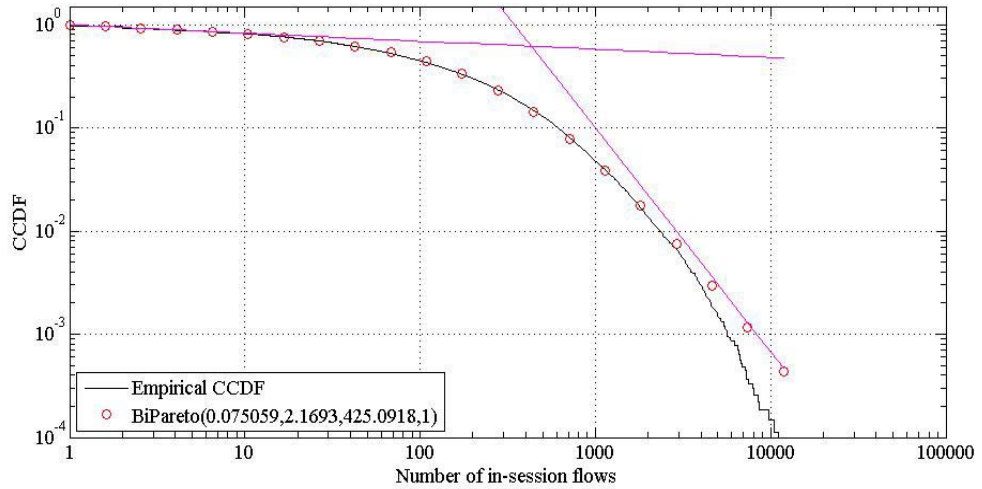


Figure 6.11: CCDF of number of Web flows per session. Theoretical vs. empirical data

- the duration and scale of the trace;
- the client arrivals volume;
- the number of flows generated per user initiated session;
- the corresponding flow sizes;

Our traffic generator ensures that the synthetic trace follows the respected models and provides data that are easy to manipulate, process, and incorporate in any simulation study.

The synthetic trace generator was implemented as a matlab module. It can be used either to simulate one of the modeled variables (e.g. number of flows per session) or to simulate variables that are not directly modeled, such as, total traffic per hour or average throughput per hour. The inputs required by the synthetic trace generator are the following:

1. Trace length as number of intervals of length  $L$ ;

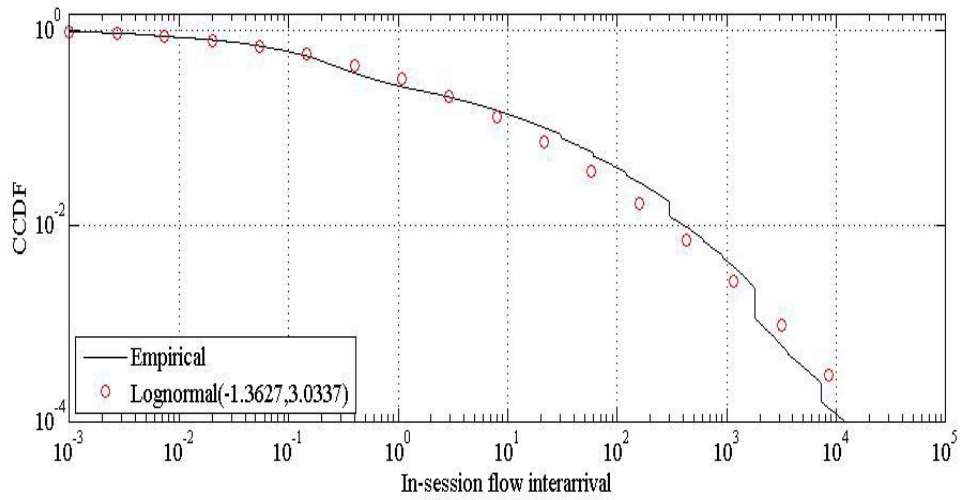


Figure 6.12: CCDF of theoretical vs. actual flow inter-arrival times

2. Duration (L) of each interval in seconds;
3. Number of sessions for each interval to be simulated (in our case fixed from real trace);
4. Parameters of the lognormal model for flow interarrivals within a session;
5. Parameters of the biPareto model for num of flows per session;
6. Parameters of the biPareto model for flow sizes;

Note that a reasonable choice for the duration of each interval of the trace is the time during which the arrival rate of the Poisson process is constant.

A high level view of the implementation of our synthetic trace generator consists of the following steps:

1. For each time interval 'i' (1,N);

Generate session interarrival times based on the exponential distribution;

- Transform session interarrival times to session arrival times;
- 2. For each session 's' generated during (1);
  - Generate number of flows ('n') through biPareto;
  - Generate 'n' flow interarrival times through lognormal;
  - Transform flow interarrival times to flow arrival times;
  - Generate 'n' flow sizes through biPareto model;
  - End(2);
  - End(1);
- 3. Based on each flow's size and arrival, calculate throughput time series

Using the synthetic trace generator, we simulated 179 hours of synthetic trace (as many as the hours of the real trace) triggering all of the modelled variables in order to calculate average throughput per hour. The interval length was set to 1 hour and the parameters for the corresponding distributions were the ones described in Section 6.2. Notice that our synthetic trace generator does not model or makes any assumptions about the duration of the flow. We only model the arrival time for each flow and not the number of intervals during which the flow was active. In order to get a rough estimate of the average throughput per hour, we assume for both the synthesized and the real data, that the flow ends at the same time interval that it starts and that all of its traffic is transferred during the corresponding time interval.

As figure 6.13 shows, the synthetic trace follows the trends observed in the actual data capturing the variability of the throughput for through the tracing period. Overallly the shape of the throughput that was computed through the synthetic trace has the same shape with the actual one, exhibiting peaks and

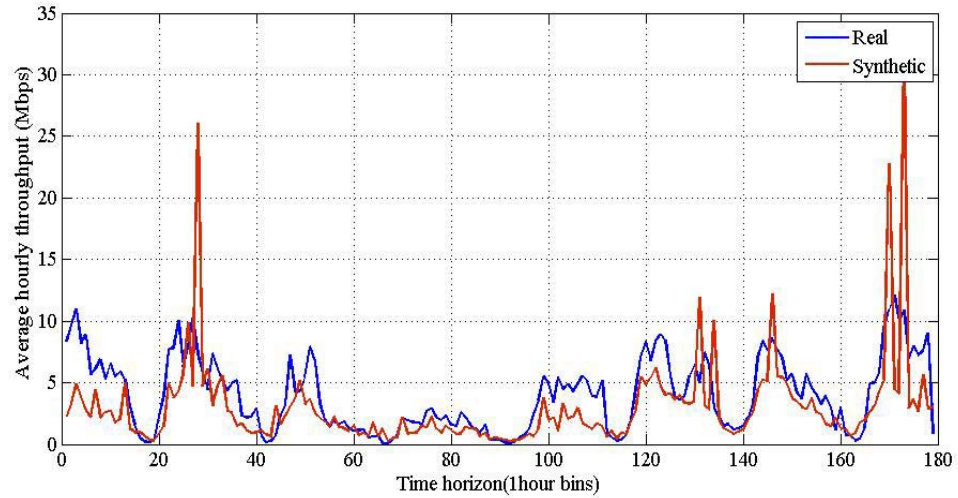


Figure 6.13: Average throughput per hour. Synthetic vs. empirical data

intervals of low utilization at the same hours. There are however some hours where the synthetic throughput is higher than the real, the synthetic generator underestimates the actual throughput. To get a rough estimate of the deviation of the synthetic trace from the actual one in terms of average throughput per hour, we calculated for interval 'i' the residual value of the throughput as:  $\text{Residual throughput}(i) = \text{Real throughput}(i) - \text{Synthetic throughput}(i)$ . The mean residual value over all intervals of the simulated period was 0.8Mbps. To get an essence of this deviation, consider that the actual flows that are modeled and simulated correspond to all flows of UNC WLAN captured at the egress router over a tracing period of 8 days. The bandwidth of the backbone link that connects UNC's WLAN to the egress router is 100Mbps.



## 6.6 Discussion

In [11] it was shown the proposed parametric models for the session- and flow-level variables can also be used to model the corresponding variables at the AP-level apart from the aggregate network. For example, focusing on hotspot APs, it was shown that the number of flows per session for a specific AP follow a biPareto distribution.

### 6.6.1 Generating synthetic traces

As shown in chapter 5 of [1], the synthetic traces produced using the synthetic trace generator described in 6.5 can be used as input in simulations of various 802.11 scenarios in Ns-2. When compared to other publicly used models, our parametric models generate traffic that is more similar to the empirical data in terms of various metrics, such as, delay, jitter, and throughput.

Apart from our parametric models, synthetic traces were generated using various other models. These models are summarized in table 6.3. Note that when a fixed number is used for flow sizes this number is the mean flow size in the empirical trace.

The empirical trace to which all synthetic traces are compared is a file that contains the session arrival timestamp, the AP at which the corresponding session started, the arrival of each in-session flow and the corresponding flow size. To generate synthetic flow sizes drawn from a specific distribution, MLE was first used to derive the parameters of the corresponding distribution and then using these parameters, values that follow that distribution were produced using a random number generator. In both the Pareto-Uniform and Fixed-Uniform synthetic traces, the flow arrival times are derived from a uniform distribution from the

Model	Flow size	Flow arrival
<b>biPareto-Lognormal</b>	biPareto	Lognormal
<b>biPareto-Lognormal-AP</b>	biPareto	Lognormal
<b>Pareto-Empirical</b>	Pareto	As in empirical trace
<b>Pareto-Uniform</b>	Pareto	Uniform
<b>Fixed-Empirical</b>	Fixed	as in empirical trace
<b>Fixed-Uniform</b>	Fixed	Uniform
<b>Lognormal-Weibul</b>	Lognormal	Weibul
<b>Fixed-Fixed</b>	Fixed	Fixed

Table 6.3: Models used to generate synthetic traces

interval  $[0, T]$ , where  $T$  is the duration of the empirical trace. The flow sizes in the Pareto-Uniform scenario were derived using a pareto distribution while the Fixed-Uniform scenario includes flow sizes that are fixed, equal to the mean flow size found in the trace. The Lognormal-Weibul scenario is based on the models proposed in [49] by Meng *et al.*, in which the flow interarrival times follow a Weibul distribution in hourly basis and flow sizes a lognormal distribution. The parameters of the Weibul distribution were determined using Maximum Likelihood Estimates for each *hour-of-day* of the empirical trace. Finally, [1] makes use of some naive models (e.g., Fixed-Fixed scenario) that have been widely used in performance analysis studies and simulations concerning wireless network protocols.

### 6.6.2 Simulation testbed

In order to evaluate the performance of all aforementioned models, synthetic traces were generated using each one of these models and were fed as input in

simulations scenarios in Ns-2. The testbed that is simulated is a WLAN with three wireless clients associated at the same AP and for wired clients connected through a router to the internet. Multiple concurrent connections are initiated by source nodes to reproduce the simulated traffic where a round-robin flow assignment process is used to associate clients with flows. For the case of naive models, flows of fixed size and fixed arrival time were used based on a simple CBR traffic generator. Apart from the traffic generated by each simulated model, there is no other background traffic in any of the simulated scenarios.

### 6.6.3 Benchmarks

For the evaluation of the performance of the various parametric models, a set of benchmarks were used. Note that these benchmarks were not directly addressed by our models and were employed to characterize the performance of IEEE802.11 APs under real-life network conditions. The benchmarks used were:

- throughput
- jitter
- delay
- goodput
- hourly aggregate traffic sent from wired clients

Note that unlike throughput that takes into account all the data transferred in the transport layer, goodput only considers the amount of bytes delivered from the transport layer to the application layer. The term delay is used to denote the average packet delay within a flow while jitter expresses the variability of the delay.

#### 6.6.4 Evaluation

Comparing all synthetic traces generated by the models included in table 6.3 through average flow delay, it is shown in [ ] that our set of parametric models captures more efficiently the trend observed in the actual data with Meng’s models being also accurate. Other naive models based on fixed flow sizes or on flow sizes drawn from a pareto distribution deviate significantly from the actual data. The proposed set of parametric models also generates synthetic traffic that is closer to actual data both in terms of average flow delay and average flow jitter when compared to Meng’s approach and the naive models mentioned in [49].

## CHAPTER 7

### Conclusions

Wireless networks grow in response to the increasing demand for wireless access. In the same time, more features are added to them to enable support for more demanding applications and efficient management of their resources. Knowledge of the wireless traffic demand in terms of the application types used is crucial to performing network overprovisioning. Moreover performance analysis studies that are intended to reveal weaknesses of wireless networks and propose new mechanisms, such as, load balancing and admission control to improve their performance, call for realistic models of key network elements. Along the direction of improving the performance of wireless networks, traffic load forecasting estimates can be used by APs to not only better manage their traffic demand but also advice clients to associate with the appropriate APs to better utilize their local resources. Such predictions can be used to reduce the energy spendings at the client side, improve the capacity utilization of wireless LANs, and better load balance the traffic.

In this work, we designed a number of forecasting algorithms based on the knowledge acquired from the modeling task, and evaluated their performance on the hotspots of a large production wireless network. We also extended these algorithms at the client level and compared their performance at the AP level. Our ultimate goal is to design and develop admission control, capacity planning and load balancing tools incorporating these forecasting mechanisms.

We also provide a detailed multi-level application based characterization of a campus wide WLAN from the perspective of the network, client, and AP. Our results can be employed to support better admission control and AP selection mechanisms, indicate usage trends, and guide per-application traffic modeling efforts. We identify application usage patterns across clients and APs. We also explore how wireless channel dynamics affect user behavior and compare application traffic mix of the infrastructure studied with other wired and wireless infrastructures.

We evaluated the performance of a set of parametric models for flow and session related variables intended to describe traffic demand of a large scale WLAN. Most of the modeling efforts have been on the AP-level. The shift to sessions and flows has gained two important advantages: sessions at an AP can mask the network-related dependencies that are not important in a range of applications and simulation environments, such as, brief transitions from one AP to another due to a transient behavior of the signal, and exhibit statistical properties (such as stationarity) that make them amenable to modeling. We also explored how these models perform when applied to model flow related variables for Web. Finally we implemented and evaluated the performance of a synthetic trace generator based on the parametric models proposed.

## CHAPTER 8

### Future work

As far as AP traffic forecasting is concerned, we intend to explore how forecasting performs in finer time scales. Short-term forecasting (e.g., next minute) can assist in designing more energy-efficient clients. Moreover, long-term forecasting is essential for capacity planning and understanding the evolution of the wireless traffic and networks. We also aim to simulate load balancing mechanisms that facilitate these forecasting algorithms to evaluate their impact on the performance of the network.

We also intend to extend our application-based characterization of wireless demand at the transport layer by using heuristics and statistical clustering techniques to profile clients based on their application characteristics and roaming patterns, and contrast the results using data from different wireless environments. Another issue that also requires significant consideration is how mobility affects both application usage patterns across clients and the performance of applications with real-time constraints. We are also concerned in exploring how wireless channel dynamics ,such as, packet losses, interference and MAC-layer retransmissions affect the performance of certain applications ,such as, P2P or voice and video streaming ones.

There is also a variety of open issues concerning the modeling methodology and the parametric models proposed for the flow- and session-level variables that were used to characterize wireless demand at the transport layer. In section 6

we have partially explored how the parametric models proposed for flow related variables perform when applied to Web flows. We intended to further explore, both through statistical tests and simulations, how well the proposed models scale when used to characterize flows of specific application types. Using traces from other infrastructures and other tracing periods we aim to investigate if our models persist over time and if they can be applied to a large variety of infrastructures to accurately characterized wireless demand. A further refinement of our models will consider how the population size of wireless users relates to the process of session arrivals. Some clients use the infrastructure only one or a few times and then disappear from the system, whereas others represent a more constant load. Understanding this part of the workload will make simulations more intuitive, since their input could be the number of clients and a parametric description of their access patterns. Apart from the statistical evaluation of our models we also intended to perform a system-oriented evaluation by synthesizing traffic based on the proposed models in a broadly used network simulator (e.g., Ns-2) and then compare the actual traces with the synthesized ones based on parameters that were not directly addressed by the corresponding models, such as, average throughput per flow, average delay and jitter.



## A Multiple Linear Regression

The purpose of multiple linear regression is to establish a quantitative relationship between a group of predictor variables (the columns of  $X$ ) and a response,  $y$ . This relationship is useful for

1. understanding which predictors have the greatest effect;
2. knowing the direction of the effect (i.e., increasing  $x$  increases/decreases  $y$ );
3. using the model to predict future values of the response when only the predictors are currently known;

The quantitative relationship between a group of predictor variables (the columns of  $X$ ) and a response,  $y$  can be expressed through a linear model of the form:  $y = X\beta + \varepsilon$  where:

- $y$  is an  $n$ -by-1 vector of observations;
- $X$  is an  $n$ -by- $p$  matrix of regressors;
- $\beta$  is a  $p$ -by-1 vector of parameters;
- $\varepsilon$  is an  $n$ -by-1 vector of random disturbances;

The solution to the problem is a vector,  $b$ , which estimates the unknown vector of parameters,  $\beta$  and is computed using the least squares method. This solution is expressed by the following equation:  $b = \hat{\beta} = (x^T X)^{-1} x^T y$

---

<sup>2</sup>Main source used: [96]

## B Least Squares Method

The method of least squares, also known as regression analysis, is used to model numerical data obtained from observations by adjusting the parameters of a model so as to get an optimal fit of the data. The best fit is characterized by the sum of squared residuals having the minimum possible value with the residual being the difference between an observed value and the value given by the model.

### B.1 Problem Statement

The objective consists of adjusting the parameters of a model function so as to best fit a data set. Consider the case of a data sample that consists of  $m$  points (data pairs)  $(x_i, y_i)$ , for  $i=1, m$ , where  $x_i$  is an independent variable and  $y_i$  is a dependent variable whose value is found through observations. Also assume that the proposed model for the independent variable  $y_i$  is:  $f(x_i, \beta)$ , where the  $n$  adjustable parameters are held in the vector  $\beta$ . Target of the least squares method is to find those parameter values for which the model “best” fits the data. The least squares method tries to define the parameters that best fit the sample data by minimizing the *sum of the squared residuals* denoted as:  $S = \sum_{i=1}^m r_i^2$  A residual is defined as the difference between the values of the dependent variable and the model and is denoted by the following equation:  $r_i = y_i - f(x_i, \beta)$  An example of a model is that of the straight line. Denoting the intercept as  $a$  and the slope as  $\beta$  then the corresponding model function is given by:  $f(x_i, \beta) = a + \beta x_i$ .

A data point may consist of more than one independent variable. For an example, when fitting a plane to a set of height measurements, the plane is a function of two independent variables,  $x$  and  $z$ , say. In the most general case

there may be one or more independent variables and one or more dependent variables at each data point.

## B.2 Solving the least squares problem

Least squares problems fall into two categories, linear and non-linear. The linear least squares problem has a closed form solution, but the non-linear problem has to be solved by iterative refinement; at each iteration the system is approximated by a linear one, so the core calculation is similar in both cases.

The minimum of the sum of squares is found by setting the gradient to zero. Since the model contains  $n$  parameters there are  $n$  gradient equations.

$$\frac{ds}{d\beta_j} = 2 \sum_i r_i \frac{dr_i}{d\beta_j} = 0, j = 1 \dots n$$

and since:

$$r_i = y_i - f(x_i, \beta)$$

the gradient equations become:

$$-2 \sum_i \frac{df(x_i, \beta)}{d\beta_j} r_i = 0, j = 1 \dots n$$

The gradient equations apply to all least squares problems. Each particular problem requires particular expressions for the model and its partial derivatives.

The system is a linear one when when the model comprises a linear combination of the parameters and can be expressed as:

$$f(x_i, \beta) = \sum_{j=1}^{j=n} x_{ij} \beta_j.$$

The coefficients  $x_{ij}$  can be either constants or functions of the independent variable  $x_i$ .

For the case of non-linear system, there is no closed solution to a non-linear least squares problem. Instead, initial values must be chosen for the parameters.

Then, the parameters are refined iteratively, that is, the values are obtained by successive approximation.

---

<sup>1</sup>Main source used: [95, 94]

## C Grubb's test

The Grubbs test, also known as the maximum normalized residual test, can be used to test for outliers in a univariate data set. Note that this test assumes that the actual data are normally distributed. Grubbs test detects one outlier at a time so in order to remove multiple outlying values, Grubbs test must be applied in an iterative manner by removing each time the current outlier. This process should be repeated until no outliers are detected. More formally, the Grubbs test can be defined as follows:

$H_0$ : There are no outliers in the data.

$H_a$ : There is at least one outlier in the data.

Significance level:  $\alpha$ .

The statistic test used for the detection of outliers is expressed through the following equation:

$$G = \frac{\max(|X_i| - \mu)}{s}$$

where  $\mu$  and  $s$  are the sample mean and standard deviation of the data. That is, the Grubbs test statistic is the largest absolute deviation from the sample mean in units of the sample standard deviation.

The hypothesis  $H_0$  of no outliers is rejected if:

$$G > \frac{N-1}{\sqrt{N}} \sqrt{\frac{t_{(\alpha/2N, N-2)}^2}{N-2+t_{(\alpha/2N, N-2)}^2}}$$

where  $t_{(\alpha/2N, N-2)}$  is the critical value of the  $t$ -distribution with  $(N-2)$  degrees of freedom and a significance level of  $\frac{\alpha}{2N}$ .

The above is actually a combination of the following two tests:

1. The test that the minimum value is an outlier.

2. The test that the maximum value is an outlier.

In order to generate the on-side tests, the test statistic is:  $G = \frac{\bar{Y} - Y_{min}}{s}$  or  $G = \frac{Y_{max} - \bar{Y}}{s}$ .

After an outlier has been detected and removed, the same procedure is repeated for the new sample with size N-1.

---

<sup>3</sup>Main source used: [97, 98]

## D Maximum Likelihood Estimate

Assume  $x$  is a continuous random variable with pdf:  $f(x; \theta_1, \theta_2, \dots, \theta_k,)$

where  $\theta_1, \theta_2, \dots, \theta_k$  are  $k$  unknown constant parameters that need to be estimated. Conducting an experiment  $N$  times we obtain  $N$  values for the random variable  $x$  ( $x_1, x_2, \dots, x_N$ ). The likelihood function is given by:

$$L(x_1, x_2, \dots, x_N | \theta_1, \theta_2, \dots, \theta_k) = L = \prod_{i=1}^N f(x_i, \theta_1, \theta_2, \dots, \theta_k)$$

for  $i=1, \dots, N$ .

The logarithmic likelihood function is:

$$\Lambda = \ln L = \sum_{i=1}^N \ln(f(x_i, \theta_1, \theta_2, \dots, \theta_k))$$

The maximum likelihood estimators (MLE) of  $\theta_1, \theta_2, \dots, \theta_k$ , are obtained by maximizing  $L$  or  $\Lambda$ . By maximizing  $\Lambda$ , which is much easier to work with than  $L$ , the maximum likelihood estimators (MLE) of  $\theta_1, \theta_2, \dots, \theta_k$  are the simultaneous solutions of  $k$  equations such that:

$$\frac{d(\Lambda)}{d\theta_j} = 0, j = 1, 2, \dots, k$$

Even though it is common practice to plot the MLE solutions using median ranks (points are plotted according to median ranks and the line according to the MLE solutions), this is not completely accurate. As it can be seen from the equations above, the MLE method is independent of any kind of ranks. For this reason, many times the MLE solution appears not to track the data on the probability plot. This is perfectly acceptable since the two methods are independent of each other, and in no way suggests that the solution is wrong.

---

<sup>4</sup>Main source used: [99]

## E Source code for specific tasks

### E.1 QQplots with simulation envelope

```
function paramhat_vec = envelope_qqplot( FLINFO_flowsizes_D_outexcl, distribution, samples )

fs = FLINFO_flowsizes_D_outexcl(:,3);
%fs = FLINFO_flowsizes_D_outexcl;
values = length(fs);
paramhat_index = 1;
%=====
% create simulation data
%=====
s = sprintf('Create synthetic data');
switch lower(distribution)
    case {'weibull'}
        [paramhat] = wblfit( fs );
        wdata0 = wblrnd(paramhat(1,1),paramhat(1,2),values,1 );
    case {'lognormal'}
        [paramhat] = lognfit( fs );
        wdata0 = lognrnd(paramhat(1,1),paramhat(1,2),values,1 );
    case {'exponential'}
        [paramhat] = expfit( fs );
        wdata0 = exprnd(paramhat(1,1),values,1 );
    case {'extreme_value'}
        [paramhat] = evfit( fs );
        wdata0 = evrnd(paramhat(1,1),paramhat(1,2),values,1 );
    case {'generalized_extreme_value'}
        [paramhat] = gevfit( fs );
        wdata0 = gevrnd(paramhat(1,1),paramhat(1,2),paramhat(1,3),values,1 );
    case {'pareto'}
        [paramhat] = gpfit( fs );
        wdata0 = gevrnd(paramhat(1,1),paramhat(1,2),std(fs),values,1 );
    case {'gamma'}
        [paramhat] = gamfit( fs );
        wdata0 = gamrnd(paramhat(1,1),paramhat(1,2),values,1 );
    case {'bipareto'}
        [paramhat] = fitBiPareto( fs, 60, 'FS', 'FS');
        wdata0 = floor(randbipareto(500, paramhat(1,1), paramhat(1,2), paramhat(1,3), min(fs)));
        % these functions elapse at ~8ms
```



```

    otherwise
        error('Invalid distribution');
    end
    paramhat_vec(paramhat_index,:) = paramhat; paramhat_index = paramhat_index + 1;
    disp ( s );
    fl = figure;
    hold on;
    %=====
    % create simulation envelope
    %=====
    for envel = 1:samples
        switch lower(distribution)
            case {'weibull'}
                s = sprintf('\t[weibull]Generating simulation envelope: run[%d]', envel);
                %[paramhat] = wblfit( fs );
                wdata = wblrnd(paramhat(1,1),paramhat(1,2),values,1 );
            case {'lognormal'}
                s = sprintf('\t[lognormal]Generating simulation envelope: run[%d]', envel);
                %[paramhat] = lognfit( fs );
                wdata = lognrnd(paramhat(1,1),paramhat(1,2),values,1 );
            case {'exponential'}
                s = sprintf('\t[exponential]Generating simulation envelope: run[%d]', envel);
                wdata = exprnd(paramhat(1,1),values,1 );
            case {'extreme_value'}
                s = sprintf('\t[extreme_value]Generating simulation envelope: run[%d]', envel);
                wdata = evrnd(paramhat(1,1),paramhat(1,2),values,1 );
            case {'gamma'}
                s = sprintf('\t[gamma]Generating simulation envelope: run[%d]', envel);
                wdata = gamrnd(paramhat(1,1),paramhat(1,2),values,1 );
            case {'generalized_extreme_value'}
                s = sprintf('\t[generalized_extreme_value]Generating simulation envelope: run[%d]', envel);
                wdata = gevrnd(paramhat(1,1),paramhat(1,2),paramhat(1,3),values,1 );
            case {'pareto'}
                s = sprintf('\t[pareto]Generating simulation envelope: run[%d]', envel);
                wdata = gprnd(paramhat(1,1),paramhat(1,2),std(fs),values,1 );
            case {'bipareto'}
                s = sprintf('\t[bipareto]Generating simulation envelope: run[%d]', envel);
                %[paramhat] = fitBiPareto( fs, 60, 'FS', 'FS' );
                wdata = floor(randbipareto(2000, paramhat(1,1), paramhat(1,2), paramhat(1,3), min(fs)));
            otherwise

```

```

        error('Invalid distribution');
    end
    disp ( s );
    h = qqplot(wdata0,wdata);
    delete(h(2)); delete(h(3));
    set(h(1),'Color','cyan')
    paramhat_vec(paramhat_index,:) = paramhat; paramhat_index = paramhat_index + 1;
end
h = qqplot(wdata0,fs);
delete(h(2)); delete(h(3));
ah = gca;
xlim = get(ah,'XLim');
ylim = get(ah,'YLim');
% f2 = figure;
% compCopy(f1, f2);
figure(f1);
%=====
% Plot data in original scale
%=====
if( xlim(1,2) > ylim(1,2) )
    axes_max = ylim;
else
    axes_max = xlim;
end
axes_max(1,1) = 0;
ah = gca;
set(ah,'XLim', axes_max);
set(ah,'YLim', axes_max);
ideal = (0:100:axes_max(1,2));
plot(ideal,ideal,'Color','r','LineWidth',2);
xlabel('Synthetic data quantiles');
ylabel('Original data quantiles');
grid on; hold off;
%=====
% Plot data in log scale
%=====
% figure(f2);
% if( xlim(1,2) > ylim(1,2) )
%     axes_max = xlim;
% else

```

```

%     axes_max = ylim;
% end
% axes_max(1,1) = 0;
% ah = gca;
% set(ah,'XLim', axes_max);
% set(ah,'YLim', axes_max);
% ideal = [0:100:axes_max(1,2)];
% plot(ideal,ideal,'Color','r','LineWidth',4);
% set(gca,'XScale','log'); set(gca,'YScale','log');
% xlabel('Synthetic data quantiles [log scale]');
% ylabel('Original data quantiles [log scale]');
% grid on;
%=====
% Save data
%=====
% switch lower(distribution)
%     case {'weibull'}
%         savename = [ 'qqfit_orig_weibull_' int2str(samples)];
%     case {'lognormal'}
%         savename = [ 'qqfit_orig_lognormal_' int2str(samples)];
%     case {'bipareto'}
%         savename = [ 'qqfit_orig_bipareto_' int2str(samples)];
%     otherwise
%         error('Invalid distribution');
% end
%saveas( f1, [ savename '.fig']);
%saveas( f1, [ savename '.jpg']);
% savenamelog = [ savename '_log'];
%close(f1);
%saveas( f2, [ savenamelog '.fig']);
% saveas( f2, [ savenamelog '.jpg']);
% close(f2);

```

## E.2 Extract basic traffic time-series from SNMP

```
#!/usr/local/bin/perl

#
# Felix Hernandez Campos
#
# February 2005
#

# -----
# USAGE
# -----

# SCRIPT NAME: ap_data2ap_ts.pl

my ($progname) = $0 =~ /^(^\/)+$/;

my $usage=<<EOF;
Description:
  Extra basic time-series from SNMP aggregate data. Note that this script
  does not perform any interpolation (but proper counter subtracting requires
  detecting reboots, so those are handled).
Input format:
  Sorted SNMP AP aggregate data trace
  sort -s -n +1 -2 +0 -1
Output format:
  Set of time-series for each AP. Each AP has its own separate file.
  This file start with two header lines preceeded by a hash (#) symbol and
  lines of the form
  relative_poll_time [ value1 value2 ... ]
Individual time-series can be easily extracted using Unix's cut or
combined with a simple perl program. Similarly, it is straight-forward
to construct time-series at coarser granulaties with a time-series
aggregator program.

Missing values are marked with NA. Completely missing samples are not
reported.
Usage:
  $progname [options] output_prefix < tr.ap_sorted_data
Options:
```

```

    -h    Show this help message.
    -...  ...
EOF

# -----
# PRAGMAS
# -----

use strict;

# -----
# IMPORTS
# -----

#die "Environment variable \$SRC undefined or incorrect"
# if ($ENV{SRC} eq "");
#require "$ENV{SRC}/stats/cdf.pl";

# -----
# MAIN
# -----

#
# OPTIONS
#

use Getopt::Std;

my (%opts);
if (not getopts("h", \%opts) or $opts{"h"} or @ARGV != 1) {
    print "$usage";
    exit;
}

use IO::File;

my $pref = $ARGV[0];

my $polling_interval = 300;
my $polling_delta = 25; # maximum expected 15 + processing time

```

```

my $assume_wrap_around_thr = 1800; # handle big wrap around after gap

# Previous and current SNMP sample
#
# Unlike declaring a new variable for each column, this type of data
# representation pays off when many columns are handled the same way,
# since we can loop through each column
my (@prev_sample, @sample);

#
# Columns in each line of the AP aggregate trace,
# and at the same time the names of the fields in the sample arrays
#

my @columns =
( "poll_time", "ap_num", "os", "up_time", "iface_speed",
  "byte_rcv", "ucast_pkt_rcv", "non_ucast_pkt_rcv", "error_pkt_rcv",
  "discarded_pkt_rcv",
  "byte_sent", "ucast_pkt_sent", "non_ucast_pkt_sent", "error_pkt_sent",
  "discarded_pkt_sent",
  "assoc", "auth", "roamed_in", "roamed_away", "deauth");

#
# Time-Series Output
#

# Set of columns from which a time-series should be computed
# Up time is useful for cleaning up the time-series
my @ts_columns = (3, 5..$#columns);
my @non_ts_columns = (0, 1, 2, 4);

# file descriptors for each time-series file
my @ts_fds;

# column number lookup by name (more readable than by index)
my %colnum;
my $i;
for ($i = 0; $i < @columns; $i++) {
    $colnum{$columns[$i]} = $i;
}

```

```

}

my @non_assoc_columns = (3, 5..($colnum{"assoc"}-1));

#
# Sample processing
#

my ($prev_line, $line);
my $num_aps = 0;
my $num_ap_reboots = 0;

# Inconsequential Cisco bug
my $off_by_one_errors = 0;

# How often is the up time after a reboot above the polling interval?
my $late_poll_after_reboot = 0;

# Current time-series file
# Samples from the SNMP aggregate input add a new line to this file
# (except for the first sample of each AP, for which no difference with
# previous sample is possible)
my $ts_fd;

my $start_poll_time;

while (<STDIN>) {

    # e.g.
    # 1077711305 6 604351400 120974027 635617 2948 1074 689172675 520663 436 23 14271809 716 1821 35970
    # notice the backwards up time: WHY is inter-poll time 74 seconds???
    # BUG???
    # 1077711379 6 604337100 120964576 635548 2948 1074 689123285 520602 436 23 14271809 716 1821 35970
    # diff is 232 to the first one and 158 to the first one
    # BUG????
    # 1077711537 6 604381000 120977516 635645 2948 1074 689224319 520689 436 23 14271809 716 1821 35970

    $line = $_;
    @sample = split;

```

```

die "Unexpected number of columns ". (scalar @sample) .
" != ". (scalar @columns) . "\n$"if (@sample != @columns);

# Ignore completely empty polls (AP did not respond)
next if ($sample[$colnum{"iface_speed"}] eq "NA");

if ($prev_sample[$colnum{"ap_num"}] ne $sample[$colnum{"ap_num"}]) {
    #
    # New AP found
    #
    $num_aps++;

    # Close time-series file for previous AP
    $ts_fd->close if (defined $ts_fd);

    # Open new time-series file for current AP
    my $fname = ">$pref.ap-" . $sample[$colnum{"ap_num"}] . ".ts";
    $ts_fd = new IO::File($fname);
    die "cannot open $fname"if (not defined $ts_fd);

    # print headers
    $start_poll_time = $sample[$colnum{"poll_time"}];
    print $ts_fd "# Start_time: $start_poll_time\n";
    print $ts_fd "# Rel-time os ";
    my $col;
    foreach $col (@ts_columns) {
        print $ts_fd ($columns[$col] . ($col != $#columns ? " " : "\n"));
    }

    # store first sample for this AP
    @prev_sample = @sample;
} else {
    #
    # Another line for this AP
    #

    my $diff = $sample[$colnum{"polltime"}] - $prev_sample[$colnum{"polltime"}];
    if ($diff < $polling_interval - $polling_delta) {
        # ToDo: output distribution
        die "Unexpected polling interval\n$prev_line$line";
    }
}

```



```

} elseif ($diff > $polling_interval + $polling_delta) {
    # warn "Large gap between polls: $diff > "
    # . ($polling_interval + $polling_delta) . "\n$prev_line$line";
    print "[ $diff ". $sample[$colnum{"polltime"}] . " - ".
    $prev_sample[$colnum{"polltime"}] . " ]\n$prev_line$line";
}

my $updiff = &substr_with_wrap_around($sample[$colnum{"up_time"}],
    $prev_sample[$colnum{"up_time"}]) / 100; # secs.

# Reboots:
# * type 1
# 1098908538 16 20804 11000000 65690 427 2 0 0 ...
# 1098908838 16 23350 11000000 0 0 0 0 0 ...
# * type 2
# 1099270071 3 976745742 11000000 1487926193 1862631 51100 57 0 ...
# 1099270392 3 NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
# 1099270672 3 27298 11000000 0 0 0 0 0 ...

# type 1
if ($sample[$colnum{"up_time"}] / 100 < $polling_interval - $polling_delta
# type 2
or $updiff < 0) {

# Rebooted AP

    if ($sample[$colnum{"up_time"}] / 100 > $diff + $polling_delta) {
        die "Up time after reboot is above polling gap\n$prev_line$line";
    } else {
        # Just rebooted
        $num_ap_reboots++;

        &reset_prev_sample;
        &flush_sample;

        $late_poll_after_reboot++;
        if ($sample[$colnum{"up_time"}] / 100 >= $polling_interval +
            $polling_delta);
    }
}

```

```

    }

} elsif ($updiff > $diff + $polling_delta) {
    warn "Up time is above polling gap\n$prev_line$line";
    # It happens sometimes
    # 1097905821 172.29.1.233 IOS 849811564 ...
    # 1097912424 172.29.1.233 IOS 850478778 ...
    # 6603 vs 6672
    &flush_sample;

} else {
    &flush_sample;
}
}
$prev_line = $line;
}

$ts_fd->close if (defined $ts_fd);

print STDERR "Number of APs found: $num_aps\n";
print STDERR "Number of AP reboots: $num_ap_reboots\n";
print STDERR "Off by one errors: $off_by_one_errors (inconsequential)\n";
print STDERR "Late polls after reboot: $late_poll_after_reboot (careful!)\n";

sub reset_prev_sample {
    my $col;
    foreach $col (@ts_columns) {
        $prev_sample[$col] = 0;
    }
}

sub reset_prev_sample_except_uptime {
    my $col;
    foreach $col (@ts_columns) {
        next if ($col == $colnum{"up_time"});
        $prev_sample[$col] = 0;
    }
}

sub reset_prev_sample_non_assoc {

```

```

my $col;
foreach $col (@non_assoc_columns) {
    next if ($col == $colnum{"up_time"});
    $prev_sample[$col] = 0;
}
}

#
# Flush sample: invoked after the second and subsequent samples of the
# current AP
#
sub flush_sample {

    my $uptime_diff = $sample[$colnum{"up_time"}] -
    $prev_sample[$colnum{"up_time"}];

    return if ($sample[$colnum{"byte_rcv"}] eq "NA" and
    $sample[$colnum{"ucast_pkt_rcv"}] eq "NA");

    my $inter_polling = $sample[$colnum{"poll_time"}] - $start_poll_time;

    printf $ts_fd "%s ", $sample[$colnum{"poll_time"}];

    # print $ts_fd "$inter_polling ";

    print $ts_fd ($sample[$colnum{"os"}] . " ");

    # Further protection from misleading wrap-around
    # Relatively common cases for VxWorks data
    if (((($sample[$colnum{"byte_rcv"}] < $prev_sample[$colnum{"byte_rcv"}]) and
    $sample[$colnum{"ucast_pkt_rcv"}] <
    $prev_sample[$colnum{"ucast_pkt_rcv"}])) or

    ($sample[$colnum{"error_pkt_rcv"}] <
    $prev_sample[$colnum{"error_pkt_rcv"}] and
    $sample[$colnum{"discarded_pkt_rcv"}] <
    $prev_sample[$colnum{"discarded_pkt_rcv"}]))

    and $uptime_diff > 0) {
        if ($sample[$colnum{"assoc"}] < $prev_sample[$colnum{"assoc"}]) {

```

```

        &reset_prev_sample_except_uptime;
    } else {
        &reset_prev_sample_non_assoc;
    }
    warn "Assuming counter reset\n$prev_line$line";
}

warn "AP ". $sample[$colnum{"ap_num"}] . " : new interface speed (" .
$prev_sample[$colnum{"iface_speed"}] . " => ".
$sample[$colnum{"iface_speed"}] . ") \n$prev_line$line"
if ($sample[$colnum{"iface_speed"}] !=
    $prev_sample[$colnum{"iface_speed"}]);

# Unreliable
# my $current_assoc = ($sample[$colnum{"assoc"}] ne "NA" and
#     $sample[$colnum{"disassoc"}] ne "NA" ?
#     $sample[$colnum{"assoc"}] -
#     $sample[$colnum{"disassoc"}] : "NA");

my $col;
foreach $col (@ts_columns) {
    my $diff;
    if ($sample[$col] eq "NA") {
        $diff = "NA";
        # do not update prev_sample, so gap can be computed
    } else {
        $diff = &substr_with_wrap_around($sample[$col], $prev_sample[$col]);

        die "Non-sensical up time $diff\n$prev_line$line"
        if ($col == $colnum{"up_time"} and $diff > 1e6
            and $inter_polling + $polling_interval < $diff / 100);

        if ($diff < 0) {
            if ($col == $colnum{"ucast_pkt_sent"} and $diff == -1) {
                # Don't worry about the following one (common Cisco bug):
                # Negative diff between two samples (colnum=10: 12344195 - 12344196)
                # 1100158382 7 1065519773 11000000 ... 12344196 ...
                # 1100158682 7 1065549758 11000000 ... 12344195 ...
                $off_by_one_errors++;
            } else {

```

```

warn "Negative difference between two samples ("
$sample[$colnum{"os"}] .
" colnum=$col: $sample[$col] - $prev_sample[$col])\n"
. "$prev_line$line";
}

if ($sample[$colnum{"os"}] == "VxW" and not $diff == -1) {
# 340s may reset within a poll
# 1097688901 152.19.141.142 VxW 1174000 11000000 30324 1226 ...
# 1097689201 152.19.141.142 VxW 1203900 11000000 467 16 ...
# 1097689501 152.19.141.142 VxW 1233900 11000000 1163 22 ...
&reset_prev_sample_except_uptime;
$diff = $sample[$col];

} elsif ($uptime_diff > 0 and
($col == $colnum{"byte_sent"} or $col == $colnum{"byte_recv"})) {
$diff = $sample[$col] + 2**32 - 1 - $prev_sample[$col];
warn "Assuming wrap around (diff=$diff)";
$prev_sample[$col] = $sample[$col];
} else {
$diff = 0;
}

} else {
$prev_sample[$col] = $sample[$col];
}
}
print $ts_fd "$diff";
print $ts_fd ($col == $ts_columns[$#ts_columns] ? "\n" : " ");
}

# print $ts_fd "--$current_assoc\n";

foreach $col (@non_ts_columns) {
$prev_sample[$col] = $sample[$col];
}
}

# Handle subtraction with 32-bit wrap around
# substr(a,b) => a - b

```

```
sub substr_with_wrap_around {  
  my ($a, $b) = @_;  
  
  if ($a <= 2**30 and $b >= 3*2**30) {  
    return $a + 2**32 - 1 - $b  
  } else {  
    return $a - $b;  
  }  
}
```

### E.3 Process time series and handle missing values

```
#!/usr/local/bin/perl

#
# Felix Hernandez Campos
#
# February-March 2005
#

# SCRIPT NAME: ap_ts2tot_traf.pl

# -----
# USAGE
# -----

my ($progrname) = $0 =~ /([\^\|]+)$/;

my $usage=<<EOF;
Description:
    Extract total traffic time-series using interpolation to handle
    missing values. See PIMRC05 submission for more details (or read the code!).
Input format:
    Output of ap_agg2ap_ts.pl
Output format:
    Two columns:
    1. Timestamp (end of bin) in seconds
    2. Value
Usage:
    $progrname [options] < tr.ts
Options:
    -h    Show this help message.
EOF

# -----
# PRAGMAS
# -----

use strict;

# -----
```

```

# MAIN
# -----

#
# OPTIONS
#

use Getopt::Std;

my (%opts);
if (not getopts("hs:", \%opts) or $opts{"h"}) {
    print "$usage";
    exit;
}

# Polling interval (in seconds)
my $poll_int = 300;
# Safe delta to handle poll retransmissions (4 attempts every 5 seconds)
my $delta_poll = 25;

# Columns in input trace

my @columns =
    ("poll_time", "os", "up_time",
     "byte_rcv", "ucast_pkt_rcv", "non_ucast_pkt_rcv", "error_pkt_rcv",
     "discarded_pkt_rcv",
     "byte_snd", "ucast_pkt_snd", "non_ucast_pkt_snd", "error_pkt_snd",
     "discarded_pkt_snd",
     "assoc", "auth", "roamed_in", "roamed_away", "deauth", "disassoc");

# column number lookup by name (more readable than by index)
my %colnum;
my $i;
for ($i = 0; $i < @columns; $i++) {
    $colnum{@columns[$i]} = $i;
}

# Find gaps in sampling and use interpolation to reconstruct the
# values if needed (downtime/reboots are replaced by zero)

```



```

my $last_ts;

my (%poll_gap_cdf, %reboot_gap_cdf);

while (<STDIN>) {

    if (/^\#/) {
        print $_if (/^\# Start/);

    } else {

        # print $_;

        my @sample = split;

        die "ERROR: Unexpected number of columns ". (scalar @sample) .
            " != ". (scalar @columns) if (@sample != @columns);

        # Total traffic
        my $value = ($sample[$colnum{"byte_recv"}] + $sample[$colnum{"byte_sent"}]);

        my $diff = $sample[$colnum{"poll_time"}] - $last_ts;

        # Detect gaps (taking into account extra time due to retransmissions)
        if ($last_ts ne "" and $diff > $poll_int + $delta_poll) {

            my $up_time = $sample[$colnum{"up_time"}] / 100;

            if ($up_time < $diff - $delta_poll) {
                #
                # Reboot Gap => nothing really happened
                #

                print STDERR "Reboot_Gap $diff $last_ts\n";

                # Print zeros for samples in the gap

                my $zero_steps = int(($diff + $delta_poll) / $poll_int) - 1;
                my $ts = $last_ts + $poll_int;
            }
        }
    }
}

```

```

my $i;
for ($i = 0; $i <= $zero_steps - 1; $i++) {
    print "$ts 0\n";
    $ts += $poll_int;
}
die "ERROR: Unexpected end ts\n$."
    if (abs($ts - $sample[$colnum{"poll_time"}]) >= $delta_poll);

# And new sample
print ($sample[$colnum{"poll_time"}] . " $value\n");

    } else {
#
# Polling Gap => interpolate missing values
#

if ($sup_time > $diff + $delta_poll) {
    print STDERR "WARNING: up time difference is larger than polling difference\n$.";
    $sup_time = $diff;
}

my $extra_steps = int(($sup_time - $delta_poll) / $poll_int);
my $diff_steps = int(($diff + $delta_poll) / $poll_int);

print STDERR "Polling_Gap $diff $last_ts\n";

# check diff between up times and polling timestamps
if (($extra_steps + 1) > $diff_steps) {
    die "ERROR: Too many interpolation steps\n$.";

} elsif (($extra_steps + 1) == $diff_steps) {
    # expected difference => interpolate
    my $i;
    my $interpolated_value = $value / $diff_steps;
    my $ts = $last_ts + $poll_int;
    for ($i = 0; $i < $diff_steps; $i++) {
        printf "$ts %.2f\n", $interpolated_value;
        $ts += $poll_int;
    }
    die "ERROR: Unexpected end ts\n$."

```

```

    if (abs($ts - $poll_int - $sample[$colnum{"poll_time"}]) >=
        $delta_poll);

} else {
    # Could add code here to handle a reboot followed by a missing poll
    die "ERROR: Short reboot\n$-";
}
}

} elsif ($diff < 0) {
    die "ERROR: Unexpected negative different of polling timestamps\n$-";

} else {
    #
    # No Gap
    #
    print ($sample[$colnum{"poll_time"}] . " $value\n");
}

$last_ts = $sample[$colnum{"poll_time"}];
}
}

```

## E.4 Synthetic trace generator

```
% oughput_syn] = synthetic_generator_flowsizesThroughputPerInterval(179, 6,
% 0.068324, 1.6442, 304.04, 1, -1.434, 2.7667, 1.6683e-010, 0.99134, 11.784, 139,session_real.6min);
function [traffic_flow_ts,throughput_syn] = synthetic_generator_flowsizesThroughputPerInterval(N, L, alpha1,
beta1, c1, k1, mu, sigma, alpha2, beta2, c2, k2,numOfSessionArrivals);

% DESCRIPTION
% The only difference from the first version is that it calculates flow
% arrival times in a different mannter.
% Input:
%
% N : [1x1] , duration of the trace [in hours]
% L : [1x1] , duration of intervals [in minutes], where
% the Poisson arrival rate can be considered to be constant
% numOfSessionArrivals :
% Fixed from the real trace and represents the number of session per
% interval of length L.
%
% Output:

num_TI = floor(N*60/L) % number of time intervals
int_len = L*60; % interval duration in seconds

counter = 1;
counter_total_flows = 0;

counter_in_traffic_flow_ts = 1;

flow_arrivals = [];
session_arrivals = [];
traffic_flow_ts = zeros( 11000000,2);

session_vector = zeros( 1, 10000 );

for n = 1:num_TI % for each time interval
    n

    %session_vector = zeros( 1, 10000 );
    % Generate session arrivals
```

```

prev_arrival_time = 0;
count_sessions = 1;
session_arrival_rate = numOfSessionArrivals(n)/int_len;

while( (prev_arrival_time <= int_len) && (session_arrival_rate~=0) )
    % this loop runs once for each time interval
    u = rand;
    cur_inter_arival = -(1/session_arrival_rate)*log(u);

    if( prev_arrival_time <= int_len )
        session_vector( 1,count_sessions ) = cur_inter_arival;
        count_sessions = count_sessions + 1;
    end

    prev_arrival_time = max( cumsum( session_vector(1,1:count_sessions) ) );
    %prev_arrival_time = max( cumsum( session_vector ) );
end

if( session_arrival_rate~=0 )

    % temp_vec = cumsum( session_vector(1,1:count_sessions) );
    temp_vec = cumsum( session_vector );
    temp_vec = temp_vec + (n-1)*int_len;
    %clear('session_vector');

    [r,c] = size( temp_vec );

    for i=1:count_sessions

        num_flows = floor(randbipareto(1, alpha1, beta1, c1, k1));

        flow_sizes = floor(randbipareto(num_flows, alpha2, beta2, c2, k2));
        %[rr,cc] = size( flow_sizes )

        counter_total_flows = counter_total_flows + num_flows;

        flow_interarrival = lognrnd(mu,sigma,1,num_flows); % flow inter-arrival times measured in seconds
        flow_interarrival = cumsum( flow_interarrival );
        flow_interarrival = flow_interarrival + temp_vec(1,i);
    end
end

```

```

vector_len = length( flow_interarrival );

for j=1:vector_len
    traffic_flow_ts(counter_in_traffic_flow_ts,1) = flow_interarrival(1,j);
    traffic_flow_ts(counter_in_traffic_flow_ts,2) = flow_sizes(1,j);

    counter_in_traffic_flow_ts = counter_in_traffic_flow_ts + 1;
end
clear('flow_interarrival');
end
end

%counter_total_flows
%count_sessions
%break;

end

counter_in_traffic_flow_ts
counter_total_flows

disp 'Calculating Throughput timeseries';
throughput_syn = zeros(179,1);
for i=1:counter_in_traffic_flow_ts-1
    bin = floor( traffic_flow_ts(i,1)/3600 );
    bin = bin + 1;
    if( bin > 179 )
    else
        throughput_syn(bin,1) = throughput_syn(bin,1) + traffic_flow_ts(i,2);
    end
end

disp 'Processing Throughput timeseries';
for i=1:179
    throughput_syn(i,1) = ((throughput_syn(i,1)*8)/1000)/3600;
end

```

## REFERENCES

- [1] M. Papadopouli, and H. Schulzrine. “Peer-to-Peer Computing for Mobile Networks: Information Discovery and Dissemination”.
- [2] M. Karaliopoulos, M. Papadopouli, E. Raftopoulos, and H. Shen. “On scalable measurement-driven modelling of traffic demand in large WLans”, *IEEE Workshop on Local and Metropolitan Area Networks*, Princeton NJ, USA, June 10-13, 2007.
- [3] H. Jiang, A. W. Moore, Z. Ge, S. Jin, and J. Wang. “Lightweight application classification for network management”, *INM '07: Proceedings of the 2007 SIGCOMM workshop on Internet network management*, Kyoto, Japan, 2007.
- [4] G. Tzagkarakis, M. Papadopouli, and P. Tsakalides. “Singular Spectrum Analysis of Traffic Workload in a Large-Scale Wireless LAN”, *10-th ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Chania, Crete Island, Greece, October 2007.
- [5] G. Tzagkarakis, M. Papadopouli, and P. Tsakalides. “Singular Spectrum Analysis of Traffic Workload in a Large-Scale Wireless LAN”, *10-th ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Chania, Crete Island, Greece, October 2007.
- [6] I. Broustis, K. Papagiannaki, S. V. Krishnamurthy, M. Faloutsos, and V. Mhatre. “MDG: measurement-driven guidelines for 802.11 WLAN design”, *ACM International Conference on Mobile Computing and Networking (MobiCom)*, Montreal, Quebec, Canada, September 2007.
- [7] G. Bianchi, A. D. Stefano, C. Giaconia, L. Scalia, G. Terrazzino, and I. Tinnirello. “Experimental assessment of the backoff behavior of commercial IEEE802.11b network cards”, *IEEE Conference on Computer Communications (InfoCom)*, Anchorage, Alaska, USA, May 2007.
- [8] A. Kashyap, S. Ganguly, and S. R. Das. “A measurement-based approach to modeling link capacity in 802.11-based wireless networks”, *ACM International Conference on Mobile Computing and Networking (MobiCom)*, Montreal, Quebec, Canada, September 2007.
- [9] Sem C. Borst and Nidhi Hegde. “Integration of streaming and elastic traffic in wireless networks”, *IEEE Conference on Computer Communications (InfoCom)*, Anchorage, Alaska, USA, May 2007.

- [10] H. Q. Nguyen, F. Baccelli, and D. Kofman. “A stochastic geometry analysis of dense IEEE802.11 networks”, *IEEE Conference on Computer Communications (InfoCom)*, Anchorage, Alaska, USA, May 2007.
- [11] F. Campos, M. Karaliopoulos, M. Papadopouli, and, H. Shen, “Spatio-Temporal Modeling of Traffic Workload in a Campus WLAN”. *Second annual international Wireless internet Conference*, Boston, USA, August, 2006.
- [12] M. Papadopouli, E. Raftopoulos, and H. Shen. “Evaluation of short-term traffic forecasting algorithms in wireless networks”, *2nd Conference on Next Generation Internet Design and Engineering*, Valencia, Spain, 2006.
- [13] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. “Impact of human mobility on the design of opportunistic forwarding algorithms”, *IEEE Conference on Computer Communications (Infocom)*, Barcelona, Spain, April 2006.
- [14] K. Ramachandran, E. Belding, K. Almeroth, and M. Bud-dhikot. “Interference-aware channel assignment in multi-radio wireless mesh networks”, *IEEE Conference on Computer Communications (Infocom)*, Barcelona, Spain, April 2006.
- [15] H. Lundgren, K. Ramachandran, E. B. Royer, K. Almeroth, M. Benny, A. Hewatt, A. Touma, and A. J. Dosh. “Experiences from the design, deployment, and usage of the UCSB Mesh-Net testbed”, *IEEE Wireless Communications*, April 2006.
- [16] Andrew Kalafut and Abhinav Acharya and Minaxi Gupta. “A Study of Malware in PeertoPeer Networks”, *Internet Measurement Conference, ACM SIGCOMM*, Rio de Janeiro, Brazil, 2006.
- [17] Saikat Guha and Neil Daswani and Ravi Jain. “An experimental study of the Skype peer-to-peer VoIP system”, *IPTPS*, Santa Barbara, CA, USA, 2006.
- [18] K. Chebrolu, B. Raman, and S Sen. “Long-distance 802.11b links: performance measurements and experience”, *ACM International Conference on Mobile Computing and Networking (MobiCom)*, Los Angeles, California, USA, September 2006.
- [19] M. Chen and A. Zakhor. “Flow control over wireless network and application layer implementation”, *IEEE Conference on Computer Communications (InfoCom)*, Barcelona, Spain, April 2006.



- [20] S. H. Y. Wong, H. Yang, S. Lu, and V. Bharghavan. “Robust rate adaptation for 802.11 wireless networks”, *ACM International Conference on Mobile Computing and Networking (MobiCom)*, Los Angeles, California, USA, September 2006.
- [21] J. Zhou, Z. Ji, and R. Bagrodia. “TWINE: A hybrid emulation testbed for wireless networks and applications”, *IEEE Conference on Computer Communications (InfoCom)*, Barcelona, Spain, April 2006.
- [22] A. P. Jardosh, K. Mittal, K. N. Ramachandran, E. M. Belding, and K. C. Almeroth. “IQU: practical queue-based user association management for WLANs”, *ACM International Conference on Mobile Computing and Networking (MobiCom)*, Los Angeles, California, USA, September 2006.
- [23] C. E. Koksal, K. Jamieson, E. Telatar, and P. Thiran. “Impacts of channel variability on link-level throughput in wireless networks”, *ACM Sigmetrics Conference on Measurements and Modeling of Computer Systems*, Saint Malo, France, June 2006.
- [24] F. Baccelli, S. Machiraju, D. Veitch, and J. Bolot. “The role of PASTA in network measurement”, *ACM Symposium on Communications Architectures and Protocols (SigComm)*, Pisa, Italy, September, 2006.
- [25] J. Kim, S. Kim, S. Choi, and D. Qiao. “Cara: Collision-aware rate adaptation for IEEE 802.11 WLANs”, *IEEE Conference on Computer Communications (InfoCom)*, Barcelona, Spain, April 2006.
- [26] F. H. Campos, and M. Papadopouli. “A Comparative Measurement Study of the Workload of Wireless Access Points in Campus Networks”, *16th Annual IEEE International Symposium on Personal Indoor and Mobile Radio Communications*, Berlin, Germany, September, 2005.
- [27] M. Papadopouli, H. Shen, M. Spanakis. “Modeling client arrivals at access points in wireless campus-wide networks”, *14th IEEE Workshop on Local and Metropolitan Area Networks*, Chania, Crete, Greece, September 18-21, 2005.
- [28] M. Papadopouli, H. Shen, and M. Spanakis, “Characterizing the duration and association patterns of wireless access in a campus”, *11th European Wireless Conference*, Nicosia, Cyprus, April 10-13, 2005.
- [29] F. H. Campos, and M. Papadopouli. “Assessing The Real Impact of 802.11 WLANs: A Large-Scale Comparison of Wired and Wireless Traffic”, *14th IEEE Workshop on Local and Metropolitan Area Networks*, Chania, Crete, Greece, 2005.

- [30] A. W. Moore, and D. Zuev. “Internet traffic classification using bayesian analysis techniques”, *SIGMETRICS '05: Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, New York, NY, USA, 2005.
- [31] T. Karagiannis, D. Papagiannaki, and, Michalis Faloutsos. “BLINC: Multi-level Traffic Classification in the Dark”, *ACM SIGCOMM*, Philadelphia, PA, USA, August 2005.
- [32] A. Moore and K. Papagiannaki. “Toward the Accurate Identification of Network Applications”, *PAM*, Boston MA, USA, March 31 - April 01, 2005.
- [33] J. Bicket, D. Aguayo, S. Biswas, and R. Morris. “Architecture and evaluation of an unplanned 802.11b mesh network”, *ACM International Conference on Mobile Computing and Networking, (MobiCom)*, Cologne, Germany, August 2005.
- [34] C. Tudeuce, and T. Gross. “A mobility model based on wlan traces and its validation”, *IEEE Conference on Computer Communications (Infocom)*, Miami, FL, March 2005.
- [35] M. Kim, and D. Kotz. “Modeling users” mobility among wifi access points”, *WiTMeMo*, Berkeley, CA, USA, June 2005.
- [36] R. Jain, D. Lelescu, and M. Balakrishnan. “Model T: an empirical model for user registration patterns in a campus wireless lan”, , Cologne, Germany, August 2005.
- [37] I. Ramani, and S. Savage. “SyncScan: Practical fast handoff for 802.11 infrastructure networks”, *IEEE Conference on Computer Communications (Infocom)*, Miami, FL, March 2005.
- [38] S.t Biswas, and R. Morris. “Opportunistic routing in multi-hop wireless networks”, *SIGCOMM Symposium on Communications Architectures and Protocols*, Philadelphia, PA, August 2005.
- [39] K. Papagiannaki, N. Taft, Z. L. Zhang, and C. Diot. “Long-Term Forecasting of Internet Backbone Traffic”, *IEEE Trans. on Neural Networks*, September, 2005.
- [40] C. Chambers, W. Feng, S. Sahu, and D. Saha. “Measurement-based Characterization of a Collection of On-line Games”, *Internet Measurement Conference, ACM SIGCOMM*, Berkley, USA, 2005.

- [41] L. Guo and Songqing Chen and Zhen Xiao and Enhua Tan and Xiaoning Ding and Xiaodong Zhang. “Measurements, Analysis, and Modeling of BitTorrent-like Systems”, *Internet Measurement Conference, ACM SIGCOMM*, Berkley, USA, 2005.
- [42] A. Kumar, E. Altman, D. Miorandi, and M. Goyal. “New insights from a fixed point analysis of single cell IEEE802.11 WLANs”, *IEEE Conference on Computer Communications (InfoCom)*, Miami, Florida, USA, March 2005.
- [43] A. Akella, G. Judd, S. Seshan, and P. Steenkiste. “Self-management in chaotic wireless deployments”, *ACM International Conference on Mobile Computing and Networking (MobiCom)*, Cologne, Germany, August 2005.
- [44] E. P. C. Jones, L. Li, and P. A. S. Ward. “Practical routing in delay-tolerant networks”, *ACM Symposium on Communications Architectures and Protocols (SigComm)*, Philadelphia, Pennsylvania, USA, August, 2005.
- [45] M. Garetto, J. Shi, and E. W. Knightly. “Modeling media access in embedded two-flow topologies of multi-hop wireless networks”, *ACM International Conference on Mobile Computing and Networking (MobiCom)*, Cologne, Germany, August 2005.
- [46] C. Chou, S. N. Shankar, and K. G. Shin. “Achieving per-stream qos with distributed airtime allocation and admission control in ieee 802.11e wireless LANs”, *IEEE Conference on Computer Communications (InfoCom)*, Miami, Florida, USA, March 2005.
- [47] T. Nadeem, L. Ji, A. K. Agrawala, and J. R. Agre. “Location enhancement to IEEE802.11 DCF”, *IEEE Conference on Computer Communications (InfoCom)*, Miami, Florida, USA, March 2005.
- [48] S. Choi, K. Park, and C. kwon Kim. “On the performance characteristics of WLANs: revisited”, *ACM Sigmetrics Conference on Measurements and Modeling of Computer Systems*, Banff, Alberta, Canada, June 2005.
- [49] X. G. Meng, S. H. Y. Wong, Y. Yuan, and S. Lu. “Characterizing flows in large wireless data networks”, *ACM MobiCom*, New York, USA, 2004.
- [50] F. Chinchilla, M. Lindsey, and M. Papadopouli. “Analysis of wireless information locality and association patterns in a campus”, *IEEE INFOCOM*, Hong Kong, March 7-11, 2004.
- [51] T. Karagiannis and A. Broido and M. Faloutsos and Kc claffy. “Transport layer identification of P2P traffic”, *Internet Measurement Conference (IMC)*, Taormina, Sicily, Italy, October 25-27, 2004.

- [52] T. Henderson, and D. Kotz, and I. Abyzov. “The changing usage of a mature campus-wide wireless network”, *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*, Philadelphia, PA, USA, 2004.
- [53] S. Sen, O. Spatscheck, and, D. Wang. “Accurate, scalable in-network identification of p2p traffic using application signatures”, *Proceedings of the 13th conference on World Wide Web*, New York, USA, May 17, 2004.
- [54] M. Roughan, S. Sen, O. Spatscheck, and N. Duffield. “Class-of-service mapping for QoS: a statistical signature-based approach to IP traffic classification”, *IMC '04: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, Taormina, Sicily, Italy, 2004.
- [55] T. Karagiannis, A. Broido, N. Brownlee, K. Claffy, M. Faloutsos. “Is P2P dying or just hiding?”, *IEEE Globecom 2004 - Global Internet and Next Generation Networks*, Dallas, Texas, USA, 29 Nov - 3 Dec, 2004.
- [56] T. Henderson, D. Kotz, and I. Abyzov, “The changing usage of a mature campuswide wireless network”. In *Proceedings of ACM MobiCom*, Philadelphia, PA, USA, September 2004.
- [57] D. Aguayo, J. Bicket, S.t Biswas, G. Judd, and R. Morris. “Link-level measurements from an 802.11b Mesh network”, *SIGCOMM Symposium on Communications Architectures and Protocols*, August 2004.
- [58] P. Bahl, R. Chandra, and J. D. Sch. “Slotted seeded channel hopping for capacity improvement in iee 802.11 ad-hoc wireless networks”, *SIGCOMM Symposium on Communications Architectures and Protocols*, Philadelphia, PA, September 2004.
- [59] Kurt Tutschku. “A Measurement-Based Traffic Profile of the eDonkey File-sharing Service”, *PAM*, Antibes Juan-les-Pins, France, 2004.
- [60] C. Dewes, A. Wichmann, and A. Feldmann. “An analysis of Internet chat systems”, *3rd ACM SIGCOMM conference on Internet measurement*, Miami Beach, FL, USA, 2003
- [61] M. Balazinska, and P. Castro. “Characterizing mobility and network usage in a corporate wireless local-area network”, *First International Conference on Mobile Systems, Applications, and Services (iMobiSys)*, San Francisco, CA, USA, May 2003.

- [62] M. Balazinska and P. Castro, “Characterizing mobility and network usage in a corporate wireless local-area network”, *MobiSys*, San Francisco, CA, USA, May 2003.
- [63] A. Jardosh, E. M. Belding-Royer, K. C. Almeroth, and S. Suri. “Towards realistic mobility models for mobile ad hoc networks”, *ACM International Conference on Mobile Computing and Networking (MobiCom)*, San Diego, CA, September 2003.
- [64] A. Mishra, M. Shin, and W. A. Arbaugh. “An empirical analysis of the IEEE 802.11 MAC layer handoff process”, *?*, April 2003.
- [65] D. Eckhardt, and P. Steenkiste. “Measurement and analysis of the error characteristics of an in building wireless network”, *ACM Computer Communication Review*, 26(4):243-254, October 1996.
- [66] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris. “A high-throughput path metric for multi-hop wireless routing”, *ACM International Conference on Mobile Computing and Networking (MobiCom)*, San Diego, CA, September 2003.
- [67] C. Nuzman, I. Saniee, W. Sweldens, and A. Weiss. “A compound model for Tcp connection arrivals for LAN and WAN applications”, *Computer Networks*, 40(3):319-337, 2002.
- [68] D. Kotz, and K. Essien. “Analysis of a Campus-wide Wireless Network”, *Eighth Annual International Conference on Mobile Computing and Networking (MobiCom)*, September, 2002.
- [69] D. Papagiannaki, N. Taft, S. Bhattacharyya, P. Thiran, K. Salamatian, and C. Diot. “A Pragmatic definition of Elephants in Internet Backbone Traffic”, *2nd ACM SIGCOMM Workshop on Internet*, Marseille, France, November 6-8, 2002.
- [70] A. Balachandran, G. Voelker, P. Bahl, and V. Rangan. “Characterizing user behavior and network performance in a public wireless lan”, *ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems*, California, USA, June, 2002.
- [71] C. Nuzman, I. Saniee, W. Sweldens, and A. Weiss. “A compound model for Tcp connection arrivals for LAN and WAN applications”, *Computer Networks*, 40(3):319-337, 2002.

- [72] A. Balachandran, G. Voelker, P. Bahl, and V. Rangan, “Characterizing user behavior and network performance in a public wireless LAN”, *ACM Sigmetrics*, CA, June 2002.
- [73] J. Cao, W. S. Cleveland, D. Lin, and D. X. Sun. “On the nonstationarity of internet traffic”, *ACM Sigmetrics*, Cambridge, MA, USA, June 2001.
- [74] F. D. Smith, F. Hernandez-Campos, K. Jeffay, and D. Ott. “What TCP/IP protocol headers can tell us about the Web”, *ACM Sigmetrics*, June 2001.
- [75] D. Tang and M. Baker. “Analysis of a local-area wireless network”, *ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, Boston, Massachusetts, USA, Aug. 2000.
- [76] W. S. Cleveland, D. Lin, and D. X. Sun. “IP packet generation: statistical models for TCP start times based on connection-rate superposition”, *ACM Sigmetrics*, Santa Clara, CA, USA, June 2000.
- [77] A. Sang, and S. Li. “A Predictability Analysis of Network Traffic”, *INFOCOM*, Tel Aviv, Israel, March, 2000.
- [78] A. Feldmann. “Characteristics of Tcp connection arrivals”, *Self-Similar Network Traffic And Performance Evaluation (K. Park and W. Willinger, eds.)*, John Wiley & Sons, 2000.
- [79] A. Mena, and J. Heidemann. “An empirical study of real audio traffic”, *IEEE Infocom*, Tel-Aviv, Israel, March 2000.
- [80] A. Bhattacharya, and S. K. Das. “LeZi-update: an information-theoretic approach to track mobile users in PCS networks”, *ACM/IEEE International Conference on Mobile Computing and Networking*, pages 1•12, Seattle, Washington, USA, August 1999.
- [81] S. Basu, and A. Mukherjee. “Time Series Models for Internet Traffic”, *24th Conf. on Local Computer Networks*, October, 1999.
- [82] P.F. Brockwell and R.A. Davis. “Time Series: Theory and Methods”, *New York: Springer-Verlag*, New York, USA, 1998.
- [83] P. Barford, and M. E. Crovella. “Generating representative Web workloads for network and server performance evaluation”, *ACM Sigmetrics*, Madison, Wisconsin, USA, June 1998.
- [84] B. A. Mah. “An empirical model of HTTP network traffic”, *IEEE Infocom*, Kobe, Japan, April 1997.

- [85] 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”, *ACM CCR*, October, 1995.
- [86] V. Paxson. “Empirically-derived analytic models of wide-area TCP connections”, *IEEE/ACM ToN*, 2(4):316-336, August 1994.
- [87] V. Paxson and S. Floyd. “Wide-area traffic: the failure of Poisson modeling”, *ACM Sigcomm*, London, United Kingdom, August 1994.
- [88] N. K. Groschwitz, and G. C. Polyzos. “A Time Series Model of Long-Term NSFNET Backbone Traffic”, *IEEE ICC*, 1994.
- [89] Sprint Applied Research Group. <http://ipmon.sprintlabs.com/packstat/packetoverview.pl>
- [90] <http://www.tcpdump.org/>
- [91] <http://www.caida.org/tools/measurement/coralreef/>
- [92] <http://www.netlab.hut.fi/opetus/s38143/luennot/>
- [93] <http://www.itl.nist.gov/div898/handbook/apr/section1/apr172.htm>
- [94] [http://en.wikipedia.org/wiki/Least\\_squares](http://en.wikipedia.org/wiki/Least_squares)
- [95] [http://www.mathworks.com/moler/least\\_squares.pdf](http://www.mathworks.com/moler/least_squares.pdf)
- [96] <http://www.mathworks.com/access/helpdesk/help/helpdesk.html>
- [97] <http://www.itl.nist.gov/div898/software/dataplot/refman1/auxillar/grubtest.htm>
- [98] <http://www.itl.nist.gov/div898/handbook/eda/section3/eda35h.htm>
- [99] [http://www.weibull.com/LifeDataWeb/appendix\\_a\\_parameter\\_estimation.htm#mle](http://www.weibull.com/LifeDataWeb/appendix_a_parameter_estimation.htm#mle)
- [100] Roofnet is an experimental 802.11b/g mesh network in development at MIT. “<http://pdos.csail.mit.edu/roofnet/doku.php>”.