ABSTRACT

Title of Thesis: WLAN Workload Characterization

Jihwang Yeo, Master of Science, 2005

Thesis directed by: Dr. Ashok K. Agrawala

Department of Computer Science

In this dissertation, we address the problem of workload characterization in a wireless LAN (WLAN). Workload is generated by applications and users trying to carry out some of their functions. We attempt to capture such application- and user-level characteristics from the information gathered at the MAC level. Developing an understandable description of the workload requires making some abstractions at the application- and user-level. Our approach is to consider the workload in terms of "sessions", where a session is an application- and user-level sequence of exchanges. We attempt to capture the session by considering an inactive duration in the activities between a wireless end-point and the network.

We consider workload to consist of a population of sessions for which a probability distribution function can be defined. Considering this distribution function to be a mixture distribution, we attempt to find the components by using non-parametric clustering technique. As the number of types of user level activities is not likely to be very large, we expect that we can associate a distinct activity with each such component. In this work, we identify such components and analyze the traffic and protocol characteristics of each component. Moreover, we empirically show that the identified workload components can effectively represent the actual WLAN workload and its daily variations.

WLAN Workload Characterization

by

Jihwang Yeo

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
2005

TABLE OF CONTENTS

List of Figures iii

1 Introduction 1
1.1 Contributions 3

2 Related Work 5

3 WLAN Workload 7
3.1 WLAN Workload Model 7
3.2 Session Generation 9
3.3 Session Features 11
3.4 Summary 13

4 Characterizing WLAN Workload 14
4.1 Session Data 14
4.2 Characteristics of Identified Workload Types 15
4.3 Workload Description 19
4.3.1 Type **MP** [**Medium-Duration MAC Probe traffic**] 19
4.3.2 Type **BM** [**Long-Duration Broadcast/Multicast Traffic**] 20
4.3.3 Type **SM** [**Short-Duration Multicast traffic**] 21
4.3.4 Type **IT** [**Isolated (One Packet) Multicast sessions**] 21
4.3.5 Type **EP** [**Excessive MAC Probe traffic**] 21
4.3.6 Type **UG** [**User Protocol Traffic in Good Channel Condition**] 22
4.3.7 Type **SP** [**Short-Duration MAC Probe traffic**] 22
4.3.8 Type **UB** [**User Protocol Traffic in Bad Channel Condition**] 22
4.3.9 Type **PT** [**137 Port Scan**] 23
4.3.10 Type **PG** [**ICMP Ping Scan**] 23
4.4 Characterization of Daily Workload Variations 23
4.5 Summary 25

5 Conclusion 26

A The Wireless Monitoring (WM) Technique 28
A.1 WM Setup 28
A.2 Implementation of WM system 28
A.2.1 Merging multiple sniffers 28
A.2.2 Sniffer placement 29

B Clustering Technique 30
B.1 Session Data 30
B.2 Transformation and Scaling on the Data 31
B.3 Distance Measures 31
B.4 The Clustering Algorithm: AMA 32
B.5 Parameter Selection 34
B.6 Quality of Clustering 37
B.7 Stability of Clustering 39

Bibliography 42

LIST OF FIGURES

1.1 WLAN workload at a wireless host 3

3.1 Session definition (figure slightly revised from [11]) 8

3.2 Number of sessions vs. timeout values 9

3.3 Distribution of number of protocols in each session generated with 30 minute timeout. 10

4.1 Daily deviation from the reference workload 24

4.2 Daily workload change: change in number of sessions 24

B.1 Number of clusters vs. d 35

B.2 Distribution of cluster sizes: select $b = 38$ (1%) 35

B.3 Distribution of cluster variances: select $v = 15$ 36

B.4 Distribution of ICDs for the clusters with the size $> b$ and the variance $< v$ (right truncated): select $d = 7, h = 4$ 36

B.5 Distribution of clustering density (left-truncated) 38

B.6 Distribution of Folkes/Mallows index 38

Chapter 1

Introduction

With the popularity of the IEEE 802.11 [1] based wireless networks, it has become increasingly important to understand the characteristics of the wireless traffic. Many measurement studies [2, 3, 4, 5, 6, 7, 8, 9, 10] have examined traffic characteristics in wireless networks. Most of the studies have focused on characterizing wireless LAN (WLAN) usage patterns and performances, which are useful for WLAN deployment and management, and workload generation.

The goal of this work is to characterize application- and user-level WLAN workload from the actual WLAN measurement. Most of the previous measurement studies have exploited packet- [9, 10], (TCP) connection- [2, 5, 7, 8], and host-level [4, 5, 7, 8, 9] information to obtain WLAN usage characteristics. Those usage characteristics can be used for generating the workload at each corresponding level. However, those levels do not properly describe such high level characteristics as the characteristics of some related tasks that applications and users will carry out through the network. As such application/user task-level characteristics are easy to understand, we can more clearly understand the WLAN workload than at other levels. In this dissertation, we attempt to identify distinct application-level tasks from measurement traces. Distinct user-level tasks can be inferred from the distinct application-level tasks. Then, we attempt to provide the descriptions of identified tasks to understand the WLAN workload.

To describe the workload at the application/user task-level, we need some abstractions of the network activities. Our approach is to consider the workload in terms of "sessions", where a session is a sequence of exchanges which may be carried out for achieving some specific task. Note that the session is different from the 802.11 session that is the duration between association and disassociation for an AP. To be more precise, a session is defined to be a bi-directional traffic unit with the same *wireless* end-point (host) as source or destination, that is separated from other sessions by at least some timeout of *inactive duration*. By considering an inactive duration in the activities of applications and users, we can identify a session of similar activities.

Modeling at the session-level gives several advantages over those at the packet- or connection-levels: First, because a session attempts to describe similar application/user activities, it can better represent distinct traffic characteristics. Second, distinct sessions (i.e., distinct application/user-level tasks) may have different demands for the networking resources, such as bandwidth and CPU consumption. Therefore, session-level modeling can also better represent the resource demands. Finally, it can model the network workload of any protocol traffic. For example, connection-level modeling can represent only the network activities using TCP protocol, whereas session-level can model all protocols.

As a basic building block for workload characterization, we represent a session as a multidimensional feature vector, where each feature is chosen to capture the basic resource usage characteristics of the session. The population of the session is considered to be defined by a probability distribution in the feature space. Further, we expect this probability distribution to be a mixture distribution such that each component of the mixture represents one type of user level activity. To identify each component of the mixture, we adopt a non-parametric clustering methodology. We developed a clustering technique, called *Adaptive Mahalanobis-distance Algorithm* (*AMA*, in short), and applied it on the sessions. As a result, we characterized two-week campus WLAN traffic of one AP (Access Point) to identify several components that represent different workload types.

In a session, an application or a user can generate a workload for a service at a specific layer, e.g., a layer among MAC-TCP/UDP. For example, as shown in Figure 1.1, a human user using a web search engine can generate a workload at the TCP layer for HTTP service. User mobility can also generate a workload at the MAC layer for the service of probing the APs with the best signal condition. Since we exploit MAC-layer measurements, we can identify various components (represented as different clusters) that are specific to a layer among MAC-TCP/UDP. For example, from the measurement traces we identified excessive MAC Probe cluster at the MAC layer, port scanning cluster at the IP layer, and broadcast traffic cluster at the TCP/UDP layer.

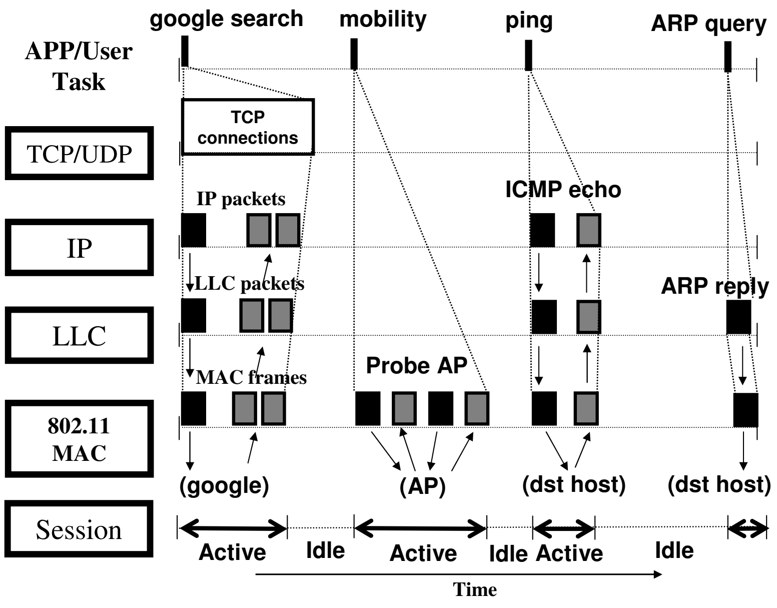

Figure 1.1: WLAN workload at a wireless host

We believe that our characterization results can be effectively used for analytical or simulation studies. Simulation studies can exploit our results by generating realistic workload according to the workload structure we empirically found. Moreover, the clustering methodology can also be used for WLAN deployment and management. WLAN administrators can better understand the way the user population is using the network resources.

1.1 Contributions

The contributions of my thesis work can be summarized as follows: we designed and implemented an accurate wireless monitoring technique [10, 12], and measured two weeks of WLAN traffic at the wireless MAC layer. From the measured data, we generated sessions as basic building blocks for workload characterization. We then developed a clustering algorithm (AMA), which classified the sessions into several clusters in non-parametric, unsupervised manner. We showed that the algorithm successfully identified several clusters and produced stable clustering results regardless of the order of input data. For characterizing the WLAN workload, we provided an understandable description of each identified cluster. Using different measurement data, we empirically showed that the identified workload components can effectively represent the actual WLAN workload and its daily variations.

The remainder of this dissertation is organized as follows. We summarize related work in Chapter 2 and describe our WLAN workload model and sessions in Chapter 3. We present the description of characterized workload types and their uses in characterizing daily workload variations in Chapter 4. In Chapter 5, we conclude this dissertation. We also attach the descriptions of our WLAN measurement technique and clustering technique, in Appendix A and Appendix B.

Chapter 2

Related Work

There have been several measurement studies of 802.11 WLANs, and in particular, university WLANs. One of the earliest was by Tang and Baker [9], who performed a twelve-week trace of the Stanford Computer Science Department WLAN. Chinchilla *et al.* [3] traced user associations and web usage on the University of North Carolina WLAN over one month. In another recent study, Schwab and Bunt [6] characterized one-week's usage and traffic patterns on the University of Saskatchewan's WLAN. A significantly larger scale experiment in terms of duration and coverage area was conducted on the Dartmouth campus WLAN by Kotz and Essien [8]. They characterized the typical usage and traffic patterns in a university WLAN over eleven weeks.

One of the few non-academic WLANs was studied by Balachandran *et al.* , who collected traces from a well-attended ACM conference [7]. They characterized not only WLAN usage patterns, but also the workloads of user arrivals and session durations with parameterized models. Balazinska and Castro traced the WLAN of a corporate research campus over the course of four weeks [4]. They characterized user mobility and traffic loads across different access points.

Similar to many of these studies, our characterization is performed in a typical university WLAN environment: a Computer Science Department network. Rather than characterize usage patterns and performance variability at IP and the above layers, we characterize typical WLAN workload structure that consists of different workload types describing application/user-level requests.

Meng *et al.* [2] statistically characterized network flows in a large campus wireless network using a trace. They characterized the flow arrivals as a Weibull regression model. While their workload model can describe the dynamic behavior, e.g., flow arrivals, our current model focuses on the static structure of the workload. We also plan to work on the dynamic workload model for each workload type in the static structure.

Clustering has been widely used for characterizing the workload for batch and interactive computer systems in early days, e.g., [13]. More recently, McGregor *et al.* [14] applied the parametric EM clustering algorithm to classify TCP flows, which were extracted from a university (non-wireless) IP traces following Claffy's model [11]. Even though they took similar approach to ours, we note several important differences. First, our target workload is for wireless network on all the wireless networking layers. Second, we carefully model the session and its features, taking into account temporal locality (i.e., timeout), resource demands and traffic characteristics. Finally, along with the well-defined session model, our non-parametric clustering algorithm clearly discriminates among the clusters with different traffic characteristics and resource demands.

Chapter 3

WLAN Workload

In this chapter, we first describe our WLAN workload model, where the workload consists of sessions as basic building blocks. We then describe how we can generate the sessions from WLAN measurement traces. Finally, we discuss the selection of session features for discriminating the workload.

3.1 WLAN Workload Model

In general, network workload consists of the *requests* for the *services* at each networking layer [15, 16]. Applications or users can make requests for the service at specific layers for carrying out their functions. Figure 1.1 shows how applications and users can generate different types of workload (requests) in a wireless host. A human user using a web search engine can make requests for HTTP service and the requests are then propagated to the lower layers. A mobile user can generate requests at the MAC layer for the service of probing for the AP with the best signal condition. Some applications can make requests at the IP layer for ICMP echo service. Requests can also be generated by other hosts. For example, an ARP (Address Resolution Protocol) query packet that is broadcast by host can make requests at the LLC (Logical Link Control) layer for ARP responses.

To describe such application- and user-level request, we define a session to capture similar requests with which applications and users can carry out a task. In this dissertation, we define a session in similar way to how Claffy *et al.* defined a *flow* in [11]. Claffy *et al.* introduced a flow as a traffic unit that has temporal (occurring closely in time) and spatial locality (occurring between the same end-points) [11]. Similarly, we define a session in WLAN as a bi-directional traffic unit with the same *wireless* end-point (host) as source or destination, that is separated from other sessions by at least a predefined timeout of inactive duration. (Selection of the timeout value will be discussed

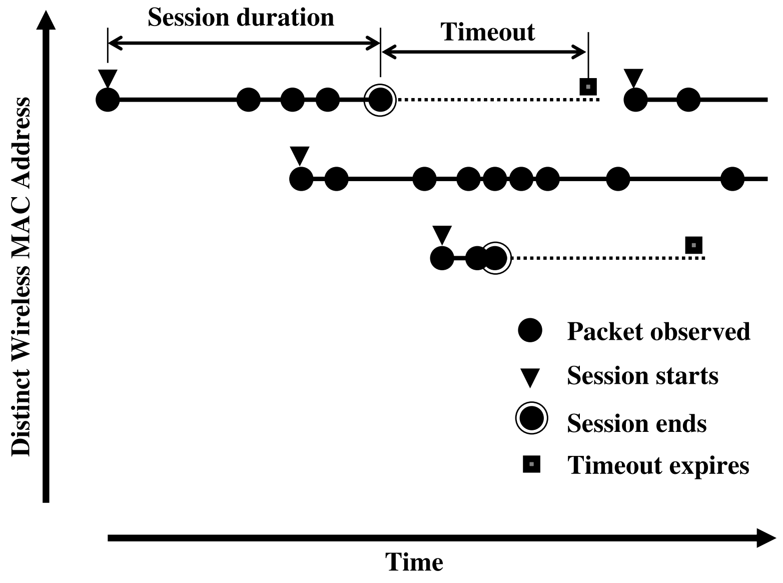

Figure 3.1: Session definition (figure slightly revised from [11])

in the next section). Since we want to model similar requests for *wireless* network service, we choose a *wireless* host as the end-point entity. Figure 3.1 illustrates the session definition. Here, we do not include the timeout in the *session duration.* Session duration is defined as the duration between the time of the first packet observed and the time of the last packet observed in a session.

We consider a session to be a basic building block for workload characterization, such that each session can describe the workload in terms of traffic characteristics and resource demands. For this purpose, we represent a session as a p-dimensional feature vector, where each feature captures the basic characteristics of a session. Note that as the users use the sessions for carrying out several types of functions, we expect that the characteristics captured by the features for similar functions to be similar.

If a session is represented as an *i.i.d.* (independent, identically distributed) random vector $X \in R^p$, we want to model the distribution of X as a multimodal distribution, and therefore as a mixture of k components. The PDF (Probability Distribution Function) of X is given as follows:

$$p(X = x) \quad = \quad \sum_{i=1}^{k} p(X = x|c_i)P(c_i), \tag{3.1}$$

where c_i is the i'th component of the mixture, $p(X = x|c_i)$ is the PDF of the i'th component, and $P(c_i)$ is the probability of the i'th component such that $0 < P(c_i) \leq 1, \sum_{i=1}^{k} P(c_i) = 1$.

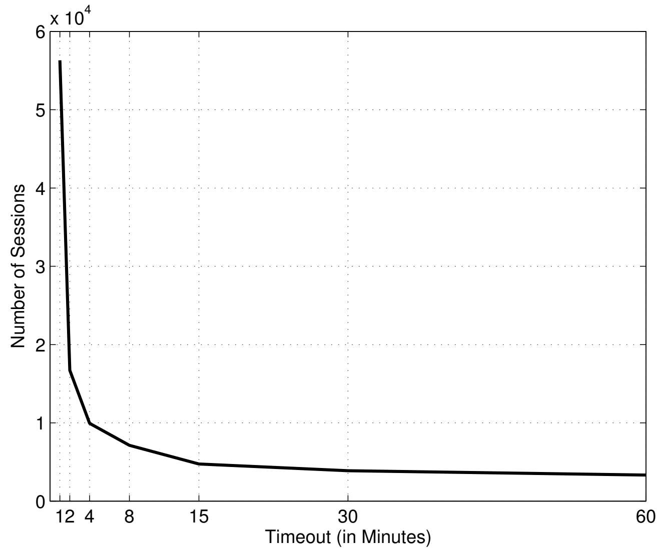

Figure 3.2: Number of sessions vs. timeout values

Instead of determining the parameters of the distribution, e.g., $k, \{P(c_i)\}_{i=1}^{k}$, we use a non-parametric clustering technique to identify the components without making any assumptions about those parameters. We apply a clustering technique, called Adaptive Mahalanobis-distance Algorithm (AMA), which combines Mahalanobis distance with several adaptive operations. Using Mahalanobis distance, we can identify the clusters of ellipsoidal shape as well as those of spherical shape in feature space. Moreover, combining the distance with several adaptive operations (e.g., creating and removing clusters dynamically to determine a proper number of clusters) enables us to effectively find the mixture components in the session data. We will describe the clustering technique in more detail in Appendix B.

3.2 Session Generation

In this chapter, we describe how to generate sessions from WLAN measurement traces. Note that the workload model introduced in the previous section makes two assumptions: the sessions are i.i.d. and the sessions represent similar requests for the WLAN network services. We generate the sessions according to the definition described in the previous section and attempt to justify these two assumptions by selecting a proper timeout value.

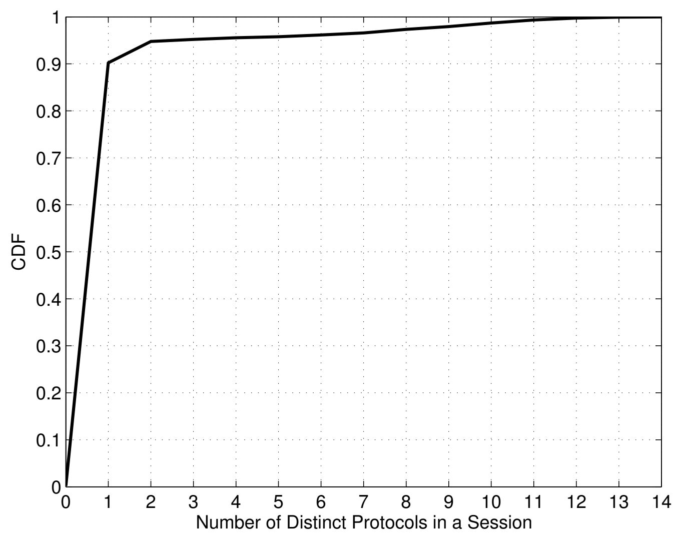

Figure 3.3: Distribution of number of protocols in each session generated with 30 minute timeout.

As shown in Figure 3.1, we generate a session for a wireless host that is identified by its wireless MAC address. For each wireless MAC address, a session for the MAC address continuously includes the packets with the MAC address as source or destination, until the wireless host of the address neither sends nor receives any packets for more than a predefined timeout.

To determine a proper timeout value, with the timeout varying over 1, 2, 4, 8, 15, 30 and 60 minutes, we generated the sessions from our two-week traffic trace according to the session definition. The result is shown in Figure 3.2. From the figure, we observe that for timeout values of 15 minutes or larger, number of sessions have become stable. We consider the sessions with the timeout values in such stable ranges to be well separated, and therefore to satisfy the *independence* assumption.

Among the stable timeout ranges ($\geq$ 15 minutes), we choose 30 minutes as the session timeout, because 30 minutes corresponds to the default web session timeout that is typically used [17, 18]. Because the web is one of the most popular applications and http traffic amounts to significant portion in wireless traffic [8, 7], 30 minute timeout can properly represent the typical inactive duration between the requests. Figure 3.3 shows the distribution of distinct number of protocols in a session that was generated with 30 minute timeout. We observe that about 90% of the sessions consist of only one protocol. This is another piece of evidence that 30 minute timeout is proper for representing similar requests.

30 minute timeout, however, may not be proper (e.g., too long) for capturing some similar application/user-level tasks. We can identify such tasks by examining the clusters which are found from the sessions with 30 minute timeout. If some cluster has too large session duration (e.g., several hours), then we may apply smaller timeout (e.g., 15 minutes) to the data in that cluster to generate new sessions.

3.3 Session Features

To use a session as a characterization building block, we need to select proper features for the workload characterization. Desirable features should properly distinguish among the sessions in terms of the traffic characteristics and resource demands.

For this purpose, we do not select any protocol information as features because the network protocol by itself does not imply the resource demands nor the traffic characteristics. For example, even different protocols (e.g., HTTP and FTP) can generate the similar traffic (e.g., large file transfer). On the other hand, the same protocol (e.g., ICMP) can generate traffic with different characteristics (e.g., echo and router solicitation). We also would not select protocol-dependent features, because we want to characterize the workload of any protocol. Even though we do not consider any protocol information for selecting the features, we note that the clusters generated by our approach to workload characterization resulted in sessions in a cluster using similar protocols.

Considering the above discussion, we select the following features: number of packets and bytes, number of MAC errors (retransmissions), number of distinct peers, and session duration. We select these *simple* features because they can be easily calculated but properly represent resource demands, such as MAC bandwidth and computing resources (CPU and memory). Moreover, they are free of protocol information and available in any protocol. Note that even though using those features we successfully characterized WLAN workload, we can add any features that can even better characterize the workload.

Here, we define these features in detail and discuss how they can discriminate among the sessions, in terms of the traffic characteristics and resource demands.

- *Number of packets* and *bytes* represent traffic volume of a session. A session with high value in these features consumes substantial bandwidth, and therefore those features distinguish the sessions with high bandwidth demands. Moreover, sessions with high traffic volume may indicate that the sessions exchange high user/application-generated traffic, rather than small machine-generated traffic. Because we notice that the two features, number of packets and bytes, are highly correlated, we use only one of them as a session feature. We choose the *number of packets* for its simplicity of calculation and representation.

- *Number of MAC errors* is obtained as the number of MAC retransmissions. MAC-level retransmissions occur when the destination host does not send a MAC ACK packet for ACK-ing the original packet, mainly due to bad signal condition or packet collisions in 802.11 WLAN. Significant MAC errors incur the waste of bandwidth and extra MAC processing, and therefore this feature can discriminate among the sessions on demands of those resources.

- *Number of distinct peers* is the number of distinct remote hosts (identified by their MAC addresses). This feature indicates how many distinct remote hosts a wireless host communicates with within a session. A session with high value in this feature may consume a large amount of memory in the wireless host for keeping the information for each peer host. Moreover, this feature can distinguish unicast traffic and broadcast traffic. For example, a wireless host typically exchanges unicast traffic with a few peers, while the traffic that is broadcast from outside the AP to the wireless side may have many distinct (source) peers because the broadcast address is shared by all hosts.

- *Session duration* is defined as the duration between the time of the first packet observed and the time of the last packet observed in a session (Figure 3.1). Excessively long session duration may lead to the overhead for reserving the memory in the wireless host for a long time. Session duration can also qualitatively distinguish the sessions. For example, sessions with very long duration, e.g., for several days, may represent housekeeping networking tasks, while short-duration sessions may represent network probing tasks, such as ICMP ping messages.

We also consider bi-directional features because due to the traffic exchange patterns a feature in different direction may have different values. We selected six features as follows: From-AP number of packets, From-AP number of MAC errors, To-AP number of packets, To-AP number of MAC errors, number of distinct peers, and session duration. Here, From-AP means the direction from the AP to the wireless host, and To-AP means the reverse direction.

3.4 Summary

In this chapter, we discussed WLAN workload model for capturing similar application/user-level tasks. For this purpose, we introduced sessions as the basic building blocks for the workload characterization. With a proper timeout value, a session can represent similar application/user-level tasks. Moreover, by selecting proper features we can distinguish the sessions in terms of traffic characteristics and resource demands.

Chapter 4

Characterizing WLAN Workload

In this chapter, we characterize the WLAN workload using the clusters obtained by the clustering techniques described in Appendix B. We identified 10 clusters from 3884 sessions of one AP over two weeks during Feb. 9 – Feb. 22, 2004. Each identified cluster represents a different *workload type* (type, in short) in terms of traffic characteristics and resource demands.

We first describe the session data and its overall characteristics in Section 4.1. We then analyze the characteristics of 10 identified workload types in Section 4.2. Next, we give a concise description in Section 4.3. Finally, in Section 4.4 we use the sessions from a two-day measurement, which were not included in the original sessions for clustering, to show that the identified workload types can characterize the workload for different measurement data. Moreover, we empirically show that the identified workload components can be effectively used for representing daily variations of WLAN workload.

4.1 Session Data

In this section, we describe the session data obtained from the two-week WLAN measurement and its overall characteristics. Our traffic trace contains per-packet information spanning all networking layers, from 802.11 MAC to TCP/UDP, as we exploited wireless monitoring techniques [12] for measurement. Detailed description on measurement set-up and methodology can be found in Appendix A.

Here, we describe our session data. From the 2-week traces, we generated 3884 sessions. We did not include 802.11 Beacon traffic in the sessions, because the request for Beacon (synchronization) service occurs at constant rate (typically 10 per second) and therefore its workload is constant. Neither did we include 802.11 Control traffic, e.g., 802.11 ACK packets, because the volume of this traffic is highly correlated with that of 802.11 Data/Management traffic. On the

Table 4.1: Statistics of Raw Session Data

	From-AP		*To-AP*			
Features	*# Packets*	*# Errors*	*# Packets*	*# Errors*	*# Peers*	*Duration (in sec)*
Mean	2738	131	505	56	7.2	3234
STD	40305	1188	6948	911	57.9	14637
Median	3	0	0	0	1	6.6
Min	1	0	0	0	1	0
Max	1877377	25416	279550	35383	2548	271174

other hand, we included 802.11 Management traffic such as MAC Probe traffic. The request for MAC Probe service is typically made on user mobility and bad signal conditions, and therefore its workload can represent WLAN-specific characteristics.

The overall statistics of the 3884 sessions are presented in Table 4.1. For each feature, sessions have significantly small mean and median values compared to the maximum, and the range of values spans several orders of magnitude. This indicates that the distribution of each feature has a long right tail, which we confirmed by examining the distributions (not shown here).

4.2 Characteristics of Identified Workload Types

In this section, we analyze traffic and protocol characteristics of 10 identified workload types. The statistics of each workload type on sessions and the features are shown in Table 4.2. From the table we can make the following observations on the traffic characteristics and resource demands of the identified types:

- The workload of type 2, 5, and 6 represent significant and unique resource demands, while the other types distinguish different traffic characteristics. Type 2 and type 6 incur significant bandwidth overhead, while type 6 leads to computing overhead for handling the excessive errors.

Table 4.2: Statistics of Identified Workload Types (we **bold-face** the types of significant resource demands)

	Session Statistics			Feature Means (in original scale)					
				From-AP		*To-AP*			
Type	*Pkt %*	*Bytes %*	*Session %*	*# Pkts*	*# Errs*	*# Pkts*	*# Errs*	*# Peers*	*Duration*
1	0.02	0.002	4.7	10.3	4.4	0	0	1	0.9 sec
2	55.8	23.6	17.8	10177	0	0	0	30.3	3.2 hr
3	0.04	0.01	39.8	3.2	0	0	0	1.003	7.6 sec
4	0.003	0.0008	10.1	1	0	0	0	1	0
5	2.2	0.3	14.8	473.7	187	0	0	1	75 min
6	40.6	75.8	3.0	26824	2862	16553	1838	30.0	4.2 hr
7	0.002	0.0003	3.6	2	1	0	0	1	0.002 sec
8	1.3	0.3	2.0	2000	795	98.2	15.9	4.2	82 min
9	0.002	0.0005	1.1	5	0	0	0	1	7 sec
10	0.002	0.0005	3.0	2	0	0	0	1	2 sec

Table 4.3: Protocols by Layers for Identified Workload Types

Types	*TCP/UDP*	*IP*	*LLC*	*MAC*
1				PROBE
2	NB-NS, NB-DGM		ARP, IAPP, STP	
3	SRVLOC	ICMP	IPv6	
4	APP-others	IGMP	IPv6	PROBE
5				PROBE
6	IMAP, HTTP, SSH	ESP		
7				PROBE
8	HTTP, SSH			PROBE
9	NB-NS	ICMP		
10		ICMP		

- A significant fraction (about 70%) of WLAN sessions have short duration less than 10 seconds.
- Most sessions, other than those in type 6 and 8, are From-AP only.

Table 4.3 and Table 4.4 show the protocol composition of each workload type, where we break down the protocols by networking layers (Table 4.3) and protocol categories (Table 4.4). We consider three protocol categories as follows:

1. *User* protocols: IMAP (Internet Message Access Protocol), HTTP, SSH, HTTP, ESP (Encapsulated Security Payloads), APP-others (unidentified or minor APP protocols).
2. *Broadcast* protocols: NB-NS (NetBios Name Service), NB-DGM (NetBios Datagram), ARP (Address Resolution Protocol), and (MAC) PROBE.
3. *Multicast* protocols: SRVLOC (Service Location Protocol), IGMP (Internet Group Management Protocol), ICMP (Internet Control Message Protocol), IPv6, IAPP (Inter-AP Protocol), and STP (Spanning Tree Protocol).

Table 4.4: Protocol by Category for Identified Workload Types

Types	*User*	*Broadcast*	*Multicast*
1		PROBE	
2		NB-NS, NB-DGM, ARP	IAPP, STP
3		PROBE	IPv6, ICMP, SRVLOC
4	APP-others	PROBE	IGMP, IPv6
5		PROBE	
6	IMAP, HTTP, SSH, ESP		
7		PROBE	
8	HTTP, SSH	PROBE	
9		NB-NS	ICMP
10			ICMP

User protocols are those, such as HTTP, SSH, FTP, etc, used for popular user applications. Broadcast protocols are basically used for "query-to-all, response-from-any" purpose through the MAC broadcast address "ff:ff:ff:ff:ff:ff". Multicast protocols are used for "query-to-some, response-from-any(-of them)" purpose through the multicast addresses, e.g., the MAC addresses starting with "01:00:5E" for IP multicasting. In the following analysis, we will show that the identified 10 workload types are properly distinguished by those three protocol categories.

Based on the information in Table 4.3 and 4.4, we observe the following:

- Because we extracted the sessions from MAC-layer measurement data, our clustering technique can identify various layer-specific workload types. In the protocol composition by layers in Table 4.3, workload of types 6 and 8 is TCP/UDP-specific, and workload of type 10 is IP-specific, respectively. We also observe that workload types 1, 5, and 7 are MAC-specific workload.

- Type 1, 5, and 7 are MAC Probe Response traffic. They do not contain corresponding MAC

Probe Request traffic. This indicates that the destination hosts of those Response packets were located far from the current AP. Because typically Probe Request from a wireless host has weaker signal than Probe Response from an AP, those Probe Request packets were not captured by our sniffers (monitoring devices). On the other hand, workload of type 8 includes both MAC Probe Request and Response, which indicates that the wireless hosts are located close to the AP and our sniffers.

- Type 6 and 8 both consist of user protocol traffic, i.e., well-known application traffic. The only difference is that type 8 has the user protocol traffic mixed with Probe traffic. This means that the difference is in the channel condition; type 6 is the user protocol traffic in a good channel condition, while the traffic of type 8 is in a slightly bad channel condition.

- Type 2 consists of broadcast and multicast traffic that use some "well-known" addresses. For example, broadcast traffic commonly uses MAC address "ff:ff:ff:ff:ff:ff" and IAPP commonly uses MAC multicast address "01:40:96:ff:ff:ff". On the other hand, the multicast traffic of Type 3 and 4 mostly uses "obscure" (i.e., not well-known) addresses. For example, IPv6 and ICMP traffic in those types uses obscure multicast addresses for neighbor discovery. IGMP traffic in those types also uses obscure multicast addresses for managing each multicast group.

4.3 Workload Description

Based on the analyses in Section 4.2, we give a concise description of each workload type. Table 4.5 summarizes these workload types.

4.3.1 Type **MP** [**Medium-Duration MAC Probe traffic**]

This workload type contains an average of a one second duration Probe Response traffic from the AP. In the sessions of this type, the wireless hosts were not in a good channel condition and had to broadcast Probe Request repeatedly (during 1 second) until they found a better AP.

Table 4.5: Summary of Identified Workload Types

Label	*Pkt %*	*Bytes %*	*Session %*	*Description*
MP	0.02	0.002	4.7	Medium-Duration MAC Probe Response Traffic
BM	55.8	23.6	17.8	Long-Duration Broadcast/Multicast Traffic
SM	0.04	0.01	39.8	Short-Duration Multicast Traffic
IT	0.003	0.0008	10.1	Isolated (one-packet) Traffic
EP	2.2	0.3	14.8	Excessive MAC Probe Response Traffic
UG	40.6	75.8	3.0	User Protocol Traffic in Good Channel Condition
SP	0.002	0.0003	3.6	Short-Duration MAC Probe Response Traffic
UB	1.3	0.3	2.0	User Protocol Traffic in Bad Channel Condition
PT	0.002	0.0005	1.1	137 Port Scan
PG	0.002	0.0005	3.0	ICMP Ping Scan

4.3.2 Type **BM [Long-Duration Broadcast/Multicast Traffic]**

The workload of this type contains broadcast/multicast traffic that uses some well-known addresses. The protocols of broadcast traffic include Net-Bios (NB-NS and NB-DGM) and ARP, which query to all hosts through MAC broadcast address "ff:ff:ff:ff:ff:ff". The multicast protocols of this type include IAPP and STP, which are used for communication between APs and switches through well-known multicast addresses, such as "01:40:96:ff:ff:ff" (IAPP) and "01:80:c2:00:00:00" (STP)[1]. Because these broadcast/multicast addresses are used by many hosts, the sessions of this type contain high traffic volume (10000 packets), a large number of peers (30), and long duration (3.2 hours). Therefore, the traffic in this type demands significant networking resources, such as bandwidth and memory.

[1]Because such IAPP/STP traffic is for inter-AP communications, APs do not have to forward such IAPP/STP traffic to the wireless side. We believe that due to proprietary AP implementation or misconfiguration, the traffic was visible to our sniffers.

4.3.3 Type **SM** [**Short-Duration Multicast traffic**]

This type contains an average of 10 second multicast traffic consisting of IPv6, ICMP, and SRVLOC protocols. IPv6 and ICMP traffic in this workload type is used for neighbor discovery through obscure (i.e., not well-known) multicast addresses. For example, if a host wants to know the link layer address of a neighbor host, it multicasts IPv6 neighbor solicitation message through the multicast address for that neighbor host. SRVLOC (Service Location Protocol) is used for automatic discovery of IP network service. For example, if a host wants to find a specific service, it sends a packet to a specific multicast address. For the multicast traffic in this workload type exploits obscure multicast addresses, the number of distinct peers, i.e., the number of distinct hosts using those addresses, is very small (nearly 1 in a session on average). Also, the sessions have low traffic volume (3.2 packets on average) and short duration (7.6 seconds on average).

4.3.4 Type **IT** [**Isolated (One Packet) Multicast sessions**]

The sessions in this workload have only one packet, and therefore we call them *isolated* sessions. Like workload type SM, this workload type contains the traffic using obscure multicast addresses. The protocols include IPv6 and IGMP. The IPv6 traffic in this workload type is for address resolution, a part of neighbor discovery. For example, if hosts want to send IPv6 packets, they query (solicit) the link layer address of the target host. The IPv6 traffic in this type consists of such neighbor solicitation packets, which were *not responded* by any neighbor hosts. Hosts use IGMP (Internet Group Management Protocol) packets to *report* their IP multicast group memberships, *query* the members of a group, and notify their *leave* from the group. The IGMP traffic in this workload type mostly consists of the "leave group" messages, which do not need follow-up packets.

4.3.5 Type **EP** [**Excessive MAC Probe traffic**]

This workload type contains an average of a 1 hour duration Probe Response traffic from the AP. In the sessions of this type, the wireless hosts had consistently sent out a large amount of Probe Request packets and received corresponding Probe Response packets (470 packets on average) for longer than 1 hour. These hosts are inferred to have been in significantly bad signal condition and therefore, had to repeatedly perform active searching for better APs.

4.3.6 Type **UG** [**User Protocol Traffic in Good Channel Condition**]

The workload of popular application protocols, such as IMAP, HTTP, SSH, esp, etc. The traffic is not mixed with MAC Probe traffic, and therefore the wireless hosts are inferred to have been in a very good channel condition. This traffic has high volume (27000 packets on average), very long duration (4.2 hours on average), and many distinct peers (30 on average), and therefore may significantly demand networking resources, such as bandwidth and memory. This traffic has From-AP and To-AP volumes well balanced, which shows typical two-way handshake patterns of popular user applications.

4.3.7 Type **SP** [**Short-Duration MAC Probe traffic**]

This workload type consists of short duration MAC Probe Response traffic with 2 packets and duration of 0.002 seconds, on average respectively. In these sessions, the wireless hosts had quickly responded to the Probe Responses from the AP. This strongly indicates that the wireless hosts may have associated with the AP.

4.3.8 Type **UB** [**User Protocol Traffic in Bad Channel Condition**]

This type of workload consists of user protocol traffic by popular applications, such as HTTP, SSH. The user traffic is mixed with MAC Probe Request/Response traffic. Those Probe traffic amounts to more than 80% in packet. Typically, the wireless STAs in bad channel condition perform active scanning searching for better APs if necessary. Therefore, the traffic of this types may come from some wireless hosts in slightly bad channel condition.

4.3.9 Type **PT** [**137 Port Scan**]

In the sessions of this type, some remote host sent exactly 5 NetBios-NS packets to port 137 during 2 seconds. These sessions are potentially malicious scanning activities, called *port 137 scan*, which aim to collect node information, e.g., a listing of any NetBios names known to that node [19]. This information may be used for the spread of Internet worm known as network.vbs [19].

4.3.10 Type **PG** [**ICMP Ping Scan**]

This type contains average 2-second duration ICMP echo request traffic. The destinations of those ICMP packets are multicast addresses, which we believe were intentionally forged. The purpose of this (potentially) malicious traffic is to discover active STAs in the network. This traffic is the evidence of a network scanning activity, called "Ping Scan". Ping Scan can be easily performed using some freely-available software, e.g., nmap.

In summary, we identified three workload types with high resource demands (BM, EP, and UG), four types of short From-AP sessions (MP, SM, IT, and SP), and two types of anomalous scanning traffic (PT and PG).

4.4 Characterization of Daily Workload Variations

In this section, we raise the question "*are these workload types also able to characterize the sessions of different measurement data?*" The characterized workload, called two-week workload, was obtained from the session data measured during Feb. 9 - Feb. 22, 2004 (two weeks). In this section, using the two-week workload we characterize the measurement data of the next two days, Feb. 23 (Monday) and Feb. 24 (Tuesday), 2004.

We first extracted "new" sessions from the measurement data of each of the two days. We then applied the same transformation and scaling described in Appendix B. To examine the deviations of the workload of new data from the two-week workload, we measured the Mahalanobis distance from each data point (session) to the identified clusters. Distribution of the Mahalanobis distances is shown in Figure 4.1. We observe that all the data points on both days have a closest

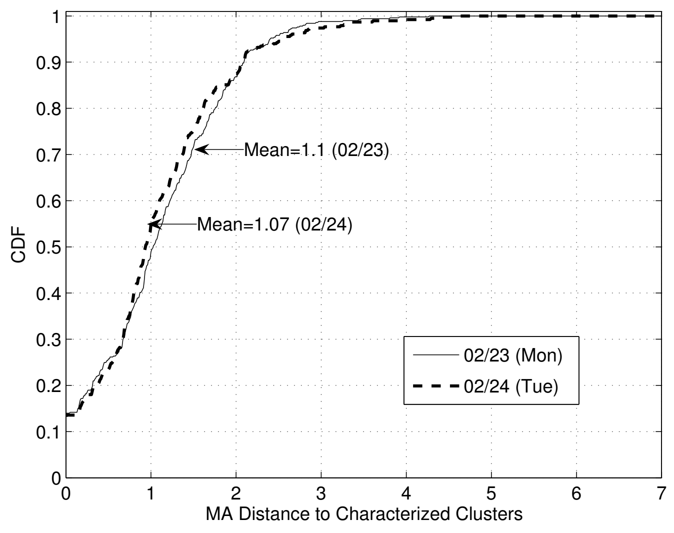

Figure 4.1: Daily deviation from the reference workload

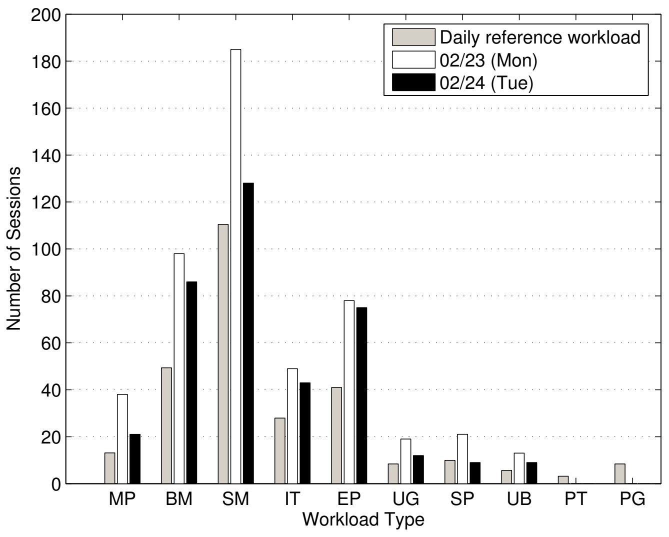

Figure 4.2: Daily workload change: change in number of sessions

cluster within 7, which is the value of d used in clustering. This indicates that there is no need for a "new" workload types to describe the "new" sessions. Moreover, the data points are as close to the clusters as one on average. This confirms that the two-week workload types can correctly characterize the workload of the two days.

Next, using the two-week workload types, we examined the workload variations over the two days. Figure 4.2 shows daily change in number of sessions in each workload type. For daily reference value, we present daily averaged number of sessions of the two-week workload for each type. From the figure, we observe that Feb. 23 and 24 both have more sessions than daily reference values in any workload type. The reason is that the two-week workload includes the workload on Saturdays and Sundays, and therefore the daily reference values are relatively small. In all workload types, Feb. 23 has more sessions than Feb. 24, because typically Monday has the highest traffic among weekdays. Finally, we note that there is no sessions for type PT and PG, on both days. This indicates that PT and PG represent the workload of anomalous traffic and therefore are not typical for every day traffic.

4.5 Summary

In this chapter, we presented 10 workload types which were found using our clustering technique. We analyzed identified different workload types in terms of traffic and protocol characteristics and provided an understandable description of each type. We also showed that these workload types can be effectively used for characterizing daily workload variations.

Chapter 5

Conclusion

In this dissertation, we addressed the problem of workload characterization in a wireless LAN (WLAN). Workload is generated by applications and users trying to carry out some of their tasks. For capturing such application/user-level tasks, we introduced "sessions" as the basic building blocks for the workload characterization. We considered the workload to consist of sessions, where a session is a sequence of exchanges for a application/user-level task. Considering the distribution of sessions to be a mixture distribution, we identified mixture components (clusters) by using non-parametric clustering technique, called AMA (Adaptive Mahalanobis distance Algorithm), and associated those clusters with distinct workload types.

From the MAC-level traces which we obtained with an accurate wireless monitoring technique [10, 12], we generated 3884 sessions. By applying the clustering technique on those sessions, we identified 10 workload types. We analyzed those workload types in terms of traffic and protocol characteristics, and provided an understandable description of each type. As a result, we identified three workload types with high resource demands, four types of short From-AP sessions, and two types of anomalous scanning traffic. Moreover, we empirically showed that the identified workload components can effectively represent the actual WLAN workload and its daily variations. We believe that our characterization results can be effectively used for analytical or simulation studies. Moreover, the clustering methodology can be used for WLAN deployment and management for exhibiting the way the user population is using the network resources.

There are several directions for future research on WLAN workload characterization. First, we can examine the *temporal workload change*, i.e., how the workload types change daily, weekly, and monthly. We can characterize the impacts of such changes on traffic characteristics and resource demands in a WLAN. Second, workload may depend on the *measurement locations.* We can characterize the workload at different measurement locations, to obtain a set of *standard*

WLAN workload types. We expect that the standard workload types can be useful for comparing the WLAN workloads at different WLAN environments. Finally, we can model dynamic behaviors of WLAN workload, such as the behaviors of packet or session arrivals. Modeling such dynamic behaviors is useful for repetitively generating different workloads as needed by the evaluation experiments.

Appendix A

The Wireless Monitoring (WM) Technique

In this section we describe our methodology, in which we use multiple sniffer devices and merge multiple datasets to improve the capture performance of the WM technique.

A.1 WM Setup

To capture wireless frames, we used three network sniffers, each comprising a PC running Linux with the 2.4.19 kernel. Each sniffer had a Prism2 chipset-based wireless network interface card; two sniffers had Demarctech DT-RWZ0-200mW-WC cards, and the third had a Linksys WPC11v3 card. To measure traffic, we used the *Ethereal* protocol analyzer (version 0.9.6) with the *libpcap* library (version 0.7). Each card was placed into 'monitor mode', which allowed the card to capture 802.11 frame information on a target channel.

The sniffers captured the first 256 bytes of each observed 802.11 frame, recording the complete view of the frame, i.e., PHY/MAC/LLC/IP/Above-IP information. PHY information, such as MAC Time and SNR (signal-to-noise ratio), can be captured using Prism2 monitor header, which is not a part of the IEEE 802.11 frame header, but is generated by the firmware of the receiving card.

A.2 Implementation of WM system

In this section, we briefly describe the WM framework, based on the techniques introduced in [10]. In that work, we demonstrated two serious drawbacks of using a single sniffer: each sniffer experiences severe loss in captured frames, and each sniffer only observes its *local view*, that is, the frames observed by one sniffer, which may differ from the AP's *global view*. Our framework aims to improve the capture performance by using multiple sniffers, placed according to SNR measurements.

A.2.1 Merging multiple sniffers

Multiple sniffers can reduce measurement loss in two ways. First, a single sniffer may not be able to observe all of the frames sent to and from a particular AP, due to radio reception and range. By using multiple sniffers, we can aggregate each sniffer's local view to create a closer approximation of the AP's global view. Second, even if a sniffer had identical radio hardware and positioning to that of an AP, it may be useful to observe the frames that the AP itself was unable to receive.

To accurately merge data from multiple sniffers, we need to be able to distinguish unique 802.11 frames for removing duplicates. We also need to prevent reordering upon merging. Reordering may occur when different sniffers observe disjoint sets of frames. For instance, if there are four frames f_{1-4} transmitted on a WLAN, and sniffer A sees f_1 and f_3, but sniffer B sees f_2 and f_4. Although each sniffer has observed their respective frames in relative order, it is impossible to use this relative order to merge the four frames. To prevent such duplication and reordering, we need to synchronize multiple sniffers' timestamps.

Our WM framework uses 802.11 Beacon frames, which are generated by the AP, as the frame of reference for all the sniffers. Beacon frames contain their own 64-bit absolute timestamps as measured by the AP, and we can therefore uniquely identify such common beacon frames in different sniffer traces. On the timestamps of such common frames, we took one of the sniffers as a reference point and used linear regression to fit the other sniffers' timestamps to the reference sniffer.

To prevent duplication and reordering, the time synchronization error (the difference between two timestamps of different sniffers for the same frame) needs to be less than *half* the minimum gap (G_{min}) between two valid IEEE 802.11 frames. In the IEEE 802.11b protocol, the minimum gap, G_{min}, can be calculated as the 192 μs (microsecond) preamble delay plus the 10 μs SIFS (Short Inter-Frame Space) and the 10 μs minimum transmission time for a MAC frame (for the case of an Acknowledgement frame) to be a total of 212 μs. Therefore, the time synchronization error needs to be less than 106 μs. Applying linear regression for each Beacon interval ($\approx$ 100ms) on 24 hours of traces from our test setup, we measured synchronization errors on the Beacon frames from another AP. We observed a maximum error of 30 μs, which is well below the 106 μs requirement. Our setup was thus suitable for measurement using multiple sniffers.

A.2.2 Sniffer placement

We used SNR measurements to place our multiple sniffers. One sniffer was placed adjacent to the AP, to be responsible for capturing the From-AP traffic and the traffic of clients near the AP. The other sniffers were placed as close as possible to the wireless clients. Assuming that clients are uniformly distributed over the coverage area, this meant placing the sniffers so that they cover as much of the AP's coverage area as possible. Generally, if we have n sniffers to place, we split the AP coverage area into n equal areas and place the sniffers in the center of mass of these areas.

To determine the AP coverage area, we first used the SNR (obtained from Prism2 header) seen in Beacon frames from the target AP to draw the *contour lines*. The AP coverage area was then determined by choosing a particular SNR contour, e.g., the 15-dB contour line.

We can refine this strategy by noting that, in an environment where multiple APs are installed, the coverage area of an AP may be reduced to the *Association Area* of the AP. The Association Area of an AP is the area at which a client will favor this AP for association compared with other APs in the area. This behavior may be device-specific and may also vary depending on whether a client has roamed to an area or has just powered on their radio. For the purposes of sniffer placement, we assume that clients will associate with the AP with the highest SNR.

Appendix B

Clustering Technique

Clustering is a non-parametric, unsupervised classification technique that does not have to determine the parameters of underlying distribution and does not exploit any external guidance. Since we do not want to make any assumptions about the parameters of the session distribution, e.g., k, $\{P(c_i)\}_{i=1}^{k}$ in Equation (3.1), non-parametric, unsupervised classification techniques, such as clustering are suitable for our WLAN characterization purpose.

In this section, we first describe our session data and transformation and scaling procedure. After discussing our distance metric, i.e., Mahalanobis distance, we give a detailed description of our algorithm, called AMA (Adaptive Mahalanobis-distance Algorithm). Followed are the discussions on the selection of the algorithm parameters. Finally, we evaluate our algorithm by examining the quality and stability of clustering results.

B.1 Session Data

In this section, we discuss the raw session data we obtained from two-week WLAN measurement.

As we discussed in Section 4.1, for each feature, sessions have significantly small mean and median values compared to the maximum, and the range of values spans several orders of magnitude (Table 4.1).

There are two problems if we use these raw data for clustering: First, similarity based on the difference of raw values may distort the actual similarity. For example, a session of 1 minute duration is as similar to a session of 2 minute duration as a session of 1000 minutes is to a session of 1001 minutes. If it is not true, we must apply some non-linear transformation (such as logarithm) to the data. Second, features have significantly different ranges to each other, which makes those features with large range dominating the clustering results. Therefore, we need to scale the data to normalize the ranges of different features. In the next section, we discuss the transformation and scaling procedures in detail.

B.2 Transformation and Scaling on the Data

To avoid the similarity distortion in each of our features, we apply the logarithm transformation for those features whose values have a range of over three orders of magnitude. From Table 4.1, we notice that all 6 features are over three orders of magnitude, and therefore are subject to the transformation. We use the equation $Y = \log_{10}(X + 1)$ for transforming a raw feature value X to a transformed value Y. Because X values are non-negative, addition of one insures non-negative Y values.

When the logarithm transformation is applied, we still have the problem of the features with large range dominating the clustering results. To handle this problem, we *scale* the values of different features into the same range. Because each feature has non-negative values in our data, we scale them into [0,10]. This procedure is called *min-max normalization* [20] and we use the following equation for scaling a logarithm-transformed value Y:

$$Z = S_{min} + S_{max} \frac{Y - Min(Y)}{Max(Y) - Min(Y)}, \tag{B.1}$$

where Z is the scaled value, and $[S_{min},\ S_{max}]$ is the target range. Here, S_{min} and S_{max} are 0 and 10 respectively.

B.3 Distance Measures

One of the key features of our algorithm is to use the Mahalanobis Distance (MD) for distance measure. In our clustering algorithm, MD is defined as the distance from a p-dimensional data point x to the center of a cluster (say c_i), as follows:

$$MD^2(x, c_i) = (x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i), \tag{B.2}$$

where p-dimensional μ_i is the center (cluster mean) of cluster c_i, and Σ_i is the covariance matrix of the already existing data points in cluster c_i. MD accounts for any correlation between the features within the cluster. Therefore, given the distance limit d as $MD(x, i) \leq d$, the cluster is maintained in an ellipsoidal shape.

In practice, Equation (B.2) should be carefully applied because in some cases Σ_i is either not available (e.g., for a cluster with one data point) or singular (e.g., for a cluster in which some data points have the same feature value). In our algorithm, in case Σ_i is not available, we instead use cI for Σ_i, where c is a constant and I is a p-dimensional identity matrix, so that the distance measure can then become the Euclidean distance. When Σ_i is singular, we have a problem that Σ_i^{-1} is not available, then we perform SVD (Singular Vector Decomposition) on Σ_i as $\Sigma_i = V \cdot \Lambda \cdot V^T$. We then replace nearly zero diagonal values in Λ by a small fraction r (e.g., 0.01) of the largest diagonal value in Λ. Thus, this procedure adjusts nearly zero minor axes of the ellipsoid to the fraction $\sqrt{r}$ (e.g., 0.1) of the major axis, keeping the cluster in a regular ellipsoidal shape.

B.4 The Clustering Algorithm: AMA

In this section, we describe the Adaptive Mahalanobis-distance Algorithm (AMA) in detail. Our clustering algorithm is non-parametric, and therefore it does not have to determine the parameters of the underlying mixture distribution. Instead, it finds each component by iteratively applying three adaptive operations: *distance-limiting*, *merging*, and *break-up*.

The algorithmic description of AMA is given in Algorithm 1. AMA iterates each pass consisting of the following three steps.

1. (Distance-limiting) The algorithm first identifies each cluster by limiting the Mahalanobis distance (MD) to a threshold d. More specifically, the algorithm either assigns each data point to the closest cluster within d, or forms a new cluster if there is no clusters within d. Whenever the point changes its cluster membership, covariance matrices of the affected clusters are recalculated.

2. (Merging) Next, the algorithm tries to obtain a compact number of clusters by merging two or more clusters that are close to each other within a threshold h in MD. Here, inter-cluster distance (ICD) is defined to be the Mahalanobis distance between cluster centers. According to our distance measures, two ICDs are calculated for a pair of clusters. Our algorithm merges two clusters if both

Algorithm 1 Adaptive Mahalanobis Distance Algorithm (AMA)

Procedure $AMA(x, dcut, d, b, v, h)$

Input: n data points (x), target density $dcut$, and four clustering parameters as follows:

- Distance-limiting thresholds (d in Mahalanobis distance),
- Break-up threshold for number of data per cluster (b),
- Break-up threshold for cluster variance (v), and
- Merging threshold for inter-cluster distance (h in Mahalanobis distance).

Output: Clustering C.

```
1: repeat
2:    (Distance-limiting)
3:    for each data point do
4:       Assign the point to a cluster if it is close from the cluster within $d$ in Mahalanobis distance from
         the cluster. Otherwise, assign the point to a new cluster having the point as the center.
5:       Mean and covariance of the cluster are updated.
6:    end for
7:    if $\rho(C)$ is the maximum so far, then $C \leftarrow$ current clusters.
8:    (Merging) Merge any two clusters whose distance between the cluster means (in both directions)
      in Mahalanobis distance is less than $h$.
9:    (Breaking up small clusters) Remove any clusters whose number of data is less than $b$.
10:   (Breaking up large clusters) Remove any clusters whose cluster variance is greater than $v$.
11: until $\rho(C) > dcut$ or no change in assignment or some maximum iteration is reached
```

the ICDs are less than h.

3. (Break-up) Finally, it improves the *quality* of the clusters by removing the clusters with their *size* less than a threshold b and the clusters with their *variance* greater than a threshold v.

For the algorithm, the size of cluster c_i, $size(c_i)$ is defined as the number of data points in c_i. The variance of c_i, denoted by $var(c_i)$ is defined as the sum of diagonal elements in Σ_i.

We define the *quality* of a cluster and a clustering (i.e., a set of clusters) by their *density*. For cluster c_i and clustering $C = \{c_i | i = 1, \dots k\}$, we define the density of c_i ($\rho(c_i)$) and the density of clustering C ($\rho(C)$) as follows:

$$\rho(c_i) = \log \frac{size(c_i)}{var(c_i)}, \rho(C) = \sum_{i=1}^{k} \frac{size(c_i)\rho(c_i)}{n}, \tag{B.3}$$

where n is the total number of data points. We take the logarithm for preventing an extremely high density of one cluster from dominating the weighted sum $\rho(C)$. In calculating $\rho(C)$, we weight $\rho(c_i)$ by $size(c_i)$ because a density change in larger clusters should more affect the quality of clustering.

To return the best quality clustering, in each pass the algorithm keeps the clustering with the maximum $\rho(C)$ so far. The algorithm terminates *either* when the maximum $\rho(C)$ reaches *dcut* (density cut) *or* after some fixed number of passes.

B.5 Parameter Selection

To select proper algorithm parameters, such as d, h, b, v, we performed quick runs of the algorithm. We varied d and ran the algorithm by only three passes (iterations) for each d. We only show the result of $d = 1, 2, \dots, 10$, because for $d > 10$ one big cluster becomes dominating the clustering. We did not apply merging and break-up operations for the runs, so that the result can only show the impacts of d.

From Figure B.1, we observe that the number of clusters significantly decreases as d increases. For $d < 5$, number of clusters is greater than 100, which is not desirable for compact characterization. Therefore, hereafter we only show the result for $d = 5, 7, 8,$ and 10. ($d = 6$ and $d = 9$ produce the similar results to $d = 5$ and $d = 10$, respectively.)

We first examine the distributions of cluster size and cluster variance, and select proper b and v that can remove "too small" and "too large" clusters. Then, we examine the distribution of

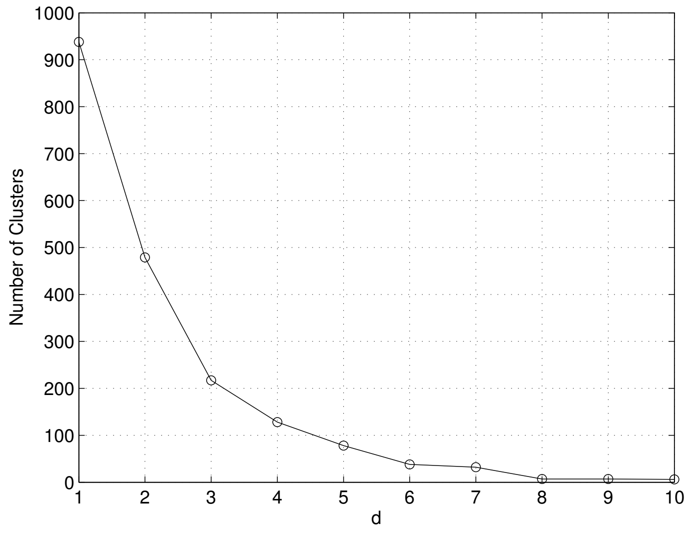

Figure B.1: Number of clusters vs. d

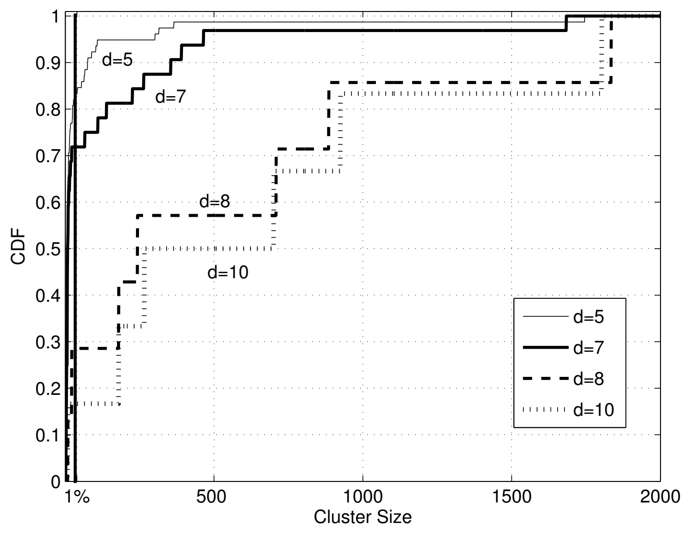

Figure B.2: Distribution of cluster sizes: select $b = 38$ (1%)

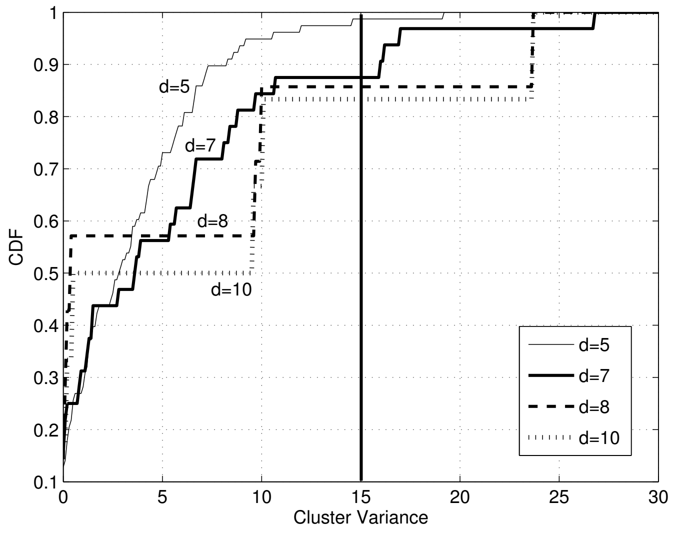

Figure B.3: Distribution of cluster variances: select $v = 15$

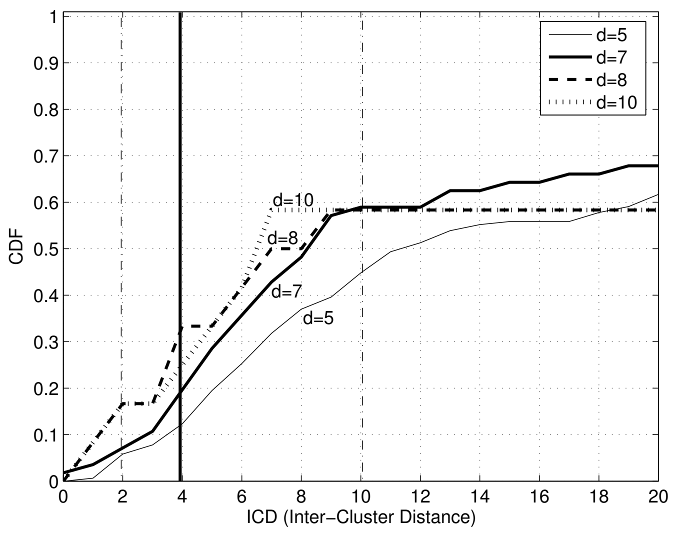

Figure B.4: Distribution of ICDs for the clusters with the size $> b$ and the variance $< v$ (right truncated): select $d = 7, h = 4$

inter-cluster distances (ICDs) to select proper d that can produce similar ICDs between all clusters. Finally, we select h so that those clusters too close each other can be merged.

For breaking up "too small" clusters, we examine the distribution of cluster sizes, as shown in Figure B.2. We observe a clear region of such "too small" clusters at around 1% of total number of points (about 38), and therefore we choose 38 as proper b value. For breaking up "too large" clusters, we examine the distribution of cluster variances, as shown in Figure B.3. We observe that for large $d(>= 7)$ there exists a clear region of large variance, e.g., that of greater than 15. We choose 15 as the proper v value.

Figure B.4 represents the distribution of ICDs. Here, we excluded those too small clusters and too large clusters with $b = 38$ and $v = 15$, because we expect those clusters will be removed by the break-up operations. High frequency of small ICDs indicates that clusters are too close to each other. Those clusters are not clearly separated, and therefore may produce unstable clustering result. On the other hand, high frequency of large ICDs means that clusters are too far apart. This indicates that the clustering consists of only those clusters with small covariance, which can hardly represent some valid clusters with large covariance.

Therefore, we choose d that can produce as similar ICDs as possible. In Figure B.4, we observe that the proper d is 7, because the majority (55%) of the ICDs exist between 2 and 10. For selecting the merging threshold h, we re-examine the ICD distribution of $d = 7$ in Figure B.4. We can identify the small ICD region, that of $<= 3$. We select $h = 4$ so that the algorithm can merge the clusters whose ICDs (in both directions) are in that small region.

B.6 Quality of Clustering

In this and the next section, we evaluate our clustering algorithm by examining the quality and stability of the clustering results. In Section B.4, we defined the quality of clustering C by the clustering density, $\rho(C)$. In this section, we discuss how the adaptive operations, i.e., distance-limiting, break-up, and merging, improve the quality of clustering. For this purpose, we ran the algorithm with $d = 7$ in five scenarios:

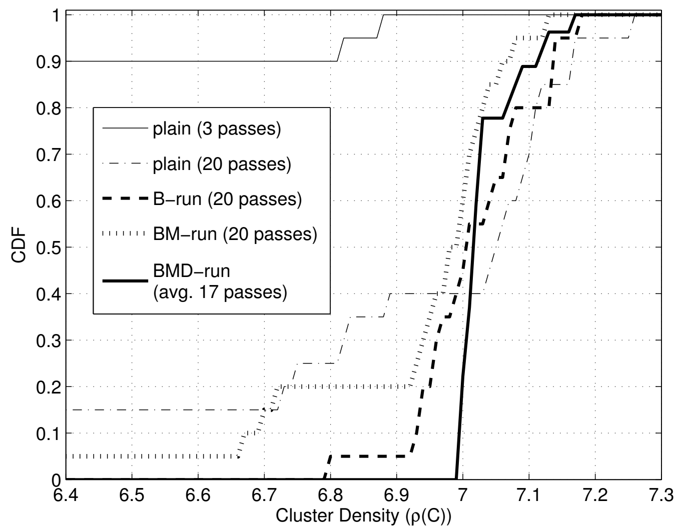

Figure B.5: Distribution of clustering density (left-truncated)

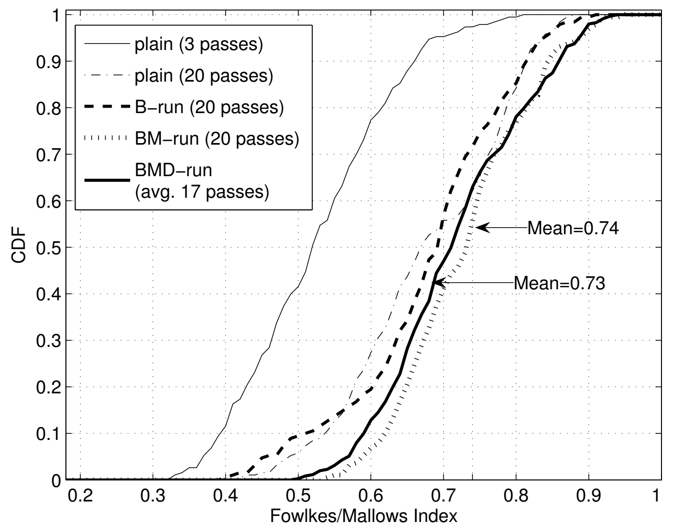

Figure B.6: Distribution of Folkes/Mallows index

1. *plain3-run*: 3 passes without break-up and merging,
2. *plain20-run*: 20 passes without break-up and merging,
3. *B-run*: 20 passes with break-up only,
4. *BM-run*: 20 passes with break-up and merging, and
5. *BMD-run*: $dcut = 7$ with break-up and merging.

The parameters previously selected ($h = 4, b = 38, v = 15$) were used, if necessary. For each scenario, we performed 20 runs with *randomly chosen order* of data points, with the same parameters. From each scenario, we obtained the distribution of $\rho(C)$, as shown in Figure B.5.

From Figure B.5, we observe that *plain20-run* produces the clusters with much higher density than *plain3-run*. The reason is that as iteration continues, distance-limiting operations gradually move the cluster centers close to the actual mean of each component. We also observe that *B-run* significantly improves the quality of clustering over the plain runs. This is because in each pass break-up operation removes low quality clusters, such as clusters with too small size or too large cluster variance.

However, *BM-run* produces worse quality than *B-run*, even though slightly better than plain runs. The reason is that merging operation only examines the ICDs and therefore sometimes it tries to merge two clusters of high density, resulting in deterioration of cluster quality. To avoid quality deterioration, we ran the algorithm with the density cut (*dcut*), without fixing the number of passes (*BMD-run*). From Figure B.5, *BMD-run* with $dcut = 7$ significantly improve the cluster quality over *BM-run*. We note that the average number of passes in *BMD-run* was 17, which means that such density cut operation can achieve high cluster quality with less computational overhead than other runs.

B.7 Stability of Clustering

Because our algorithm exploits dynamic creation and assignment of clusters on the data points coming in order, clustering result may be sensitive to the order of data points. By *stability* of clustering, we mean how similar the clustering results are for different orders of the data points.

For the measure of similarity between clusters, we use Folkes and Mallows index (FMI) [21]. For two clusterings on the same data points, FMI calculates the probability of (any) two data points belonging to the same cluster in one clustering, if those two points also belong to the same cluster in the other clustering. For clusterings I and J, $FMI(I, J)$ is given as follows:

$$FMI(I, J) = \frac{N_{I,J}}{\sqrt{N_I}\sqrt{N_J}} = \sqrt{\frac{N_{I,J}}{N_I}}\sqrt{\frac{N_{I,J}}{N_J}}, \tag{B.4}$$

where N_I (N_J) is the number of data point pairs belonging to the same cluster in clustering I (J), and $N_{I,J}$ is the number of data point pairs belonging to the same cluster in both clustering I *and* J. FMI is a probability and therefore the range is between 0 (no similarity) and 1 (perfect similarity).

To examine the stability over the different order of data points, we use the same scenarios and clustering results described in the previous section: *plain3-run*, *plain20-run*, *B-run*, *BM-run*, and *BMD-run*. Recall that for each scenario we performed 20 runs with *randomly chosen order* of data points with the same parameters. From the 20 runs in each scenario, we obtained 190 ($= \binom{20}{2}$) pairs of clusterings. Then, we calculated 190 FMIs, whose distribution for each scenario is presented in Figure B.6.

plain20-run produces more stable clusterings than *plain3-run* because with more passes distance-limiting operations move the cluster centers close to the actual centers of components. Even though *B-run* significantly improves the cluster quality over *plain20-run*, it does not produce better stability than *plain20-run*. The reason is that sometimes a "valid" cluster is broken up, and whenever it occurs distance-limiting for those points should start from the scratch.

BM-run produces most stable clusterings than other runs. Unstable clusterings are mostly due to the points around the cluster boundary that frequently change the cluster membership. Merging operations avoid this boundary situation, and therefore *BM-run* produces such highly stable clusterings. However, as pointed out in the previous section, *BM-run* may reduce the cluster quality by producing the low quality merged clusters. From Figure B.6, we observe that *BMD-run* not only achieves high-quality clusterings but also produces as stable clusterings as *BM-run*. The reason is that *BMD-run* exploits merging operations, and at the same time tries to return high-quality clustering. Again, we note that *BMD-run* produced such high-quality, stable clusterings with only 17 passes on average.

BIBLIOGRAPHY

[1] IEEE Computer Society LAN MAN Standards Committee. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.* IEEE, 1999.

[2] X. Meng, S. Wong, Y. Yuan, and S. Lu. Characterizing flows in large wireless data networks. In *Proceedings of the Tenth Annual International Conference on Mobile Computing and Networking, Philadelphia, PA*, October 2004.

[3] F. Chinchilla, M. Lindsey, and M. Papadopouli. Analysis of wireless information locality and association patterns in a campus. In *Proceedings of IEEE Infocom, Hong Kong, China*, March 2004.

[4] Magdalena Balazinska and Paul Castro. Characterizing Mobility and Network Usage in a Corporate Wireless Local-Area Network. In *1st International Conference on Mobile Systems, Applications, and Services (MobiSys)*, San Francisco, CA, May 2003.

[5] Tristan Henderson, David Kotz, and Ilya Abyzov. The changing usage of a mature campus-wide wireless network. In *Proceedings of the Tenth Annual International Conference on Mobile Computing and Networking.* ACM Press, September 2004.

[6] D. Schwab and R. Bunt. Characterising the use of a campus wireless network. In *Proceedings of IEEE Infocom, Hong Kong, China*, March 2004.

[7] A. Balachandran, G.M. Voelker, P. Bahl, and V. Rangan. Characterizing user behavior and network performance in a public wireless lan. In *Proceedings of ACM SIGMETRICS, Marina Del Rey, CA*, June 2002.

[8] David Kotz and Kobby Essien. Analysis of a campus-wide wireless network. In *Proceedings of the Eighth Annual International Conference on Mobile Computing and Networking*, pages 107–118, September 2002. Revised and corrected as Dartmouth CS Technical Report TR2002-432.

[9] D. Tang and M. Baker. Analysis of a local-area wireless network. In *Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking, Boston, MA*, August 2000.

[10] J. Yeo, M. Youssef, and A. Agrawala. A Framework for Wireless LAN Monitoring and its Applications . In *Third ACM Workshop on Wireless Security (WiSe'04), in conjunction with ACM MobiCom 2004, Philadelphia, PA*, October 2004.

[11] K. Claffy, H.-W. Braun, and G. Polyzos. A parameterizable methodology for internet traffic flow profiling. *IEEE Journal on Selected Areas in Communications*, 13(8):1481–1494, March 1995.

[12] J. Yeo, M. Youssef, and A. Henderson, T. Agrawala. An Accurate Technique for Measuring the Wireless Side of Wireless Networks. In *International Workshop on Wireless Traffic Measurements and Modeling (WiTMeMo '05), in conjunction with MobiSys '05, Seattle, Washington*, June 2005.

[13] A.-K. Agrawala, J.-M. Mohr, and R.-M. Bryant. An approach to the workload characterization problem. *Computer*, pages 18–32, 1976.

[14] Anthony McGregor, Mark Hall, Perry Lorier, and James Brunskill. Flow clustering using machine learning techniques. In *PAM*, pages 205–214, 2004.

[15] S. Raghavan, D. Vasukiammaiyar, and G. Haring. Generative networkload models for a single server environment. In *Proceedings of ACM SIGMETRICS, Nashville, Tennessee*, May 1994.

[16] R. Jain. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling.* Wiley- Interscience, New York, 1991.

[17] Lara D. Catledge and James E. Pitkow. Characterizing browsing strategies in the World-Wide Web. *Computer Networks and ISDN Systems*, 27(6):1065–1073, 1995.

[18] Huang X., Peng F., An A., and Schuurmans D. A dynamic web log session boundary detection based on statistical language modeling. *Journal of the American Society for Information Science and Technology (JASIST)*, 55(14):1290–1303, 2004.

[19] B. Alexander. Intrusion detection: Port 137 scan. In *http://www.sans.org/resources/idfaq/port_137.php*.

[20] Jiawei Han and Micheline Kamber. *Data mining: concepts and techniques.* Morgan Kaufmann Publishers Inc., 2000.

[21] E.-B. Fowlkes and C.-L. Mallows. A method for comparing two hierarchical clusterings. *Journal of the American Statistical Association*, 78(383):553–584, 1983.