In compliance with the
Canadian Privacy Legislation
some supporting forms
may have been removed from
this dissertation.

While these forms may be included
in the document page count,
their removal does not represent
any loss of content from the dissertation.

**ADAPTIVE SAMPLING METHODS FOR NETWORK
PERFORMANCE METRICS MEASUREMENT AND EVALUATION
IN MPLS-BASED IP NETWORKS**

by

Wenhong Ma

Ottawa-Carleton Institute for Electrical and Computer Engineering
Department of Systems and Computer Engineering
Faculty of Engineering
Carleton University
Ottawa, Ontario, Canada

A Thesis submitted to The Faculty of Graduate Studies
Carleton University
in partial fulfillment of the requirements for the degree of

**Master of Applied Science
in Electrical Engineering**

© Wenhong Ma 2003

National Library
of Canada

Bibliothèque nationale
du Canada

Acquisitions and
Bibliographic Services

Acquisitions et
services bibliographiques

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

ISBN: 0-612-88887-8

Our file Notre référence

ISBN: 0-612-88887-8

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

Canada

ABSTRACT

As the Internet grows in scale and complexity, the benefits of network performance measurements and monitoring are significantly increasing. Besides having a good knowledge about the behavior of the network for operational purposes, measurements can assist the traffic engineering algorithms in the optimization and dimensioning of the network by providing the feedback to operators or management systems in an efficient fashion. In selecting a network measurement and monitoring method there is a fundamental design trade-off, between overhead resource costs, and the accuracy and timeliness of the data. In order to plan a suitable sampling strategy it is therefore crucial to determine what information is needed and the desired degree of accuracy in advance. Sampling-based measurement methods provide adequate techniques for reducing the quantity of control data and attract growing interests.

The research reported in this dissertation, addresses the issue of how to carry out the sampling in an adaptive fashion, so that the accuracy for measuring the quality of service parameters (delay, loss, jitter, throughput) is better if we know something about the traffic type and traffic parameters. Our study investigates the mechanisms that can be set up to adaptively adjust the parameters of the sampling technique. Two realistic network topologies based on an MPLS-based IP networks are used to evaluate the adaptive sampling schemes for monitoring and measuring network performance metrics in relevant to voice, video and Internet data applications. Compared with conventional sampling techniques (systematic and stratified sampling), simulation results are presented to illustrate that adaptive sampling provides the potential for better monitoring, control, and management of high-performance networks with higher accuracy.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincere appreciation and gratitude to my supervisors, Dr. James Yan and Dr. Changcheng Huang for providing valuable advice and guidance whenever I need them, and for directing and supporting my work, as well as their trust and patience throughout my study. Their extensive experience and knowledge in the networking and communications fields are great technical support and precious professional resource keeping me up with the new technologies.

I also would like to thank my colleagues and friends for sharing their brilliant ideas and thoughts with me, and for the feedback they provided me. Special thanks go to Ms. Jie Zhu, Mr. Ruoyuan Zhang, Wei Yi and Chang He for their useful discussions and help.

Finally, I would like to deeply thank my family for all their support and encouragement through this effort, especially to my dear husband, Aimin for his invaluable and consistent support, offering sincere advice, and loving care.

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION	1
1.1 Motivation for this Research	1
1.1.1 Current IP Networks and Real Time Applications	1
1.1.2 The Need for Network Performance Monitoring and Measurements	3
1.2 Network Performance Measurement Metrics and Methods	5
1.2.1 Network Performance Metrics	5
1.2.1.1 Delay	5
1.2.1.2 Jitter	6
1.2.1.3 Loss	7
1.2.1.4 Throughput	8
1.2.2 Network measurement approaches	8
1.2.2.1 Active Measurement	8
1.2.2.2 Passive Measurement	9
1.2.2.3 Hybrid Monitoring Method	9
1.2.2.4 Sampling Methods	10
1.3 Previous Research Work and Problem Statement	11
1.4 Objectives of the Thesis	16
1.5 Thesis Organization	18
1.6 Thesis Contributions	18
CHAPTER 2 ADAPTIVE SAMPLING APPROACH	21
2.1 Classification of Sampling Methods	22
2.1.1 Sampling Algorithm	22
2.1.1.1 Conventional Sampling	22
2.1.2 Sampling Frequency and Interval Length	26
2.1.3 Problems of Conventional Sampling	28
2.2 Adaptive Measurement Mechanism	30
2.2.1 Adaptive Sampling Technique	30
2.2.1.1 Time Sliding Window Rate Estimator	32

2.2.1.2 Sample Size Estimation Algorithm.....	36
2.2.1.3 Embedded Monitoring Packets.....	36
2.2.2 Measurement Results.....	38
2.2.2.1 Delay and Delay Variation.....	39
2.2.2.2 Loss.....	39
CHAPTER 3 SIMULATION MODELS	41
3.1 System Model	41
3.2 Traffic Source Models	44
3.2.1 Voice Source Model	44
3.2.2 Video Source Model	46
3.2.3 Internet Data Source Model.....	48
3.3 MPLS-Based IP Network Model.....	50
3.3.1 Experimental Simple Network Model	50
3.3.2 Experimental Network Model with Competing Traffic.....	52
3.4 Performance Measurement Meter.....	52
3.5 Simulation Model Implementation	57
3.5.1 Model Translation.....	57
3.5.1.1 Network Model	57
3.5.1.2 Node Model	58
3.5.1.3 Process Model.....	59
3.5.2 Implementation Validation	60
CHAPTER 4 ANALYSIS OF SIMULATION RESULTS	62
4.1 Introduction.....	62
4.2 Performance Evaluation of the Adaptive and Systematic and Stratified Sampling Methods.....	64
4.2.1 Simple Topology for Voice	64
4.2.2 Simple Topology for Video Traffic	67
4.2.3 Simple Topology for Self-similar Traffic	70
4.2.4 Competing Traffic for Voice	73
4.2.5 Competing Traffic for Self-similar Traffic	78
4.3 Factors Impacting on the Network Performance	82

4.3.1 Effect of Traffic Utilization	82
4.3.2 Effect of Buffer Size	86
4.4 Summary of Simulation Results	89
CHAPTER 5 CONCLUSIONS	90
5.1 Conclusions.....	90
5.2 Future Work	92
REFERENCES	94

LIST OF FIGURES

Figure 2.1 Systematic sampling	23
Figure 2.2 Conventional sampling methods	26
Figure 2.3 Procedure of adaptive sampling	32
Figure 2.4 TSW rate estimator.....	35
Figure 2.5 Format of the monitoring packet	37
Figure 2.6 Adaptive random sampling.....	38
Figure 3.1 Structure of system model	42
Figure 3.2 Voice packet generation pattern	44
Figure 3.3 On-off Voice Source Model	45
Figure 3.4 Self-Similar traffic generated by a-stable traffic model	49
Figure 3.5 MPLS simple networks	51
Figure 3.6 MPLS with competing traffic.....	53
Figure 3.7 State-chart diagram of the performance measurement meter in Ingress LER	54
Figure 3.8 Arrival state flow chart diagram.....	55
Figure 3.9 Sample Size Estimation state flow chart diagram	56
Figure 3.10 Flow diagram of the performance measurement meter in Egress LER....	56
Figure 3.11 Competing traffic network model	58
Figure 3.12 Label switching router node model	59
Figure 3.13 Adaptive sampling process state model	60
Figure 4.1 Delay vs. Block size – voice traffic.....	65
Figure 4.2 Delay jitter vs. Block size – voice traffic	66
Figure 4.3 Loss ratio vs. Block size – voice traffic	67
Figure 4.4 Delay vs. Block size – video traffic.....	69
Figure 4.5 Delay jitter vs. Block size – video traffic	69
Figure 4.6 Loss ratio vs. Block size – video traffic	70

Figure 4.7 Delay vs. Block size – self-similar traffic	71
Figure 4.8 Delay jitter vs. Block size – self-similar traffic.....	71
Figure 4.9 Loss ratio vs. Block size – self-similar traffic	73
Figure 4.10 Delay vs. Competing traffic load (block size = 300)	74
Figure 4.11 Delay jitter vs. Competing traffic load (block size = 300).....	75
Figure 4.12 Delay vs. Competing traffic load (block size = 500)	75
Figure 4.13 Delay jitter vs. Competing traffic load (block size = 500).....	76
Figure 4.14 Loss ratio vs. Competing traffic load (block size = 300)	77
Figure 4.15 Loss ratio vs. Competing traffic load (block size = 500)	77
Figure 4.16 Delay vs. Competing traffic load (block size = 300)	79
Figure 4.17 Delay jitter vs. Competing traffic load (block size = 300).....	80
Figure 4.18 Delay vs. Competing traffic load (block size = 500)	80
Figure 4.19 Delay jitter vs. Competing traffic load (block size = 500).....	81
Figure 4.20 Loss ratio vs. Competing traffic load (block size = 300)	81
Figure 4.21 Loss ratio vs. Competing traffic load (block size = 500)	82
Figure 4.22 Delay mean vs. Utilization – voice traffic.....	83
Figure 4.23 Delay STD vs. Utilization – voice traffic.....	84
Figure 4.24 Delay mean vs. Utilization – self-similar traffic	84
Figure 4.25 Delay STD vs. Utilization – self-similar traffic	85
Figure 4.26 Delay mean vs. Buffer size – voice traffic	87
Figure 4.27 Delay STD vs. Buffer size – voice traffic	87
Figure 4.28 Delay mean vs. Buffer size – self-similar traffic.....	88
Figure 4.29 Delay STD vs. Buffer size – self-similar traffic.....	88

TABLE OF ACRONYMS

DS	Differentiated Services
FEC	Forwarding Equivalence Classes
FIFO	First In First Out
FSM	Finite State Machine
FSN	Fractional Stable Noise
FTP	File Transport Protocol
GPS	Global Positioning System
HTML	HyperText Markup Language
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPDV	IP Packet Delay Variation
ISP	Internet Service Provider
ITU	International Telecommunications Union
LDP	Label Distribution Protocol
LER	Label Edge Router
LSP	Label Switched Path
LSR	Label Switch Router
MPEG	Moving Picture Experts Group
MPLS	Multi-Protocol Label Switching
OPNET	OPTimized Network Engineering Tool
QoS	Quality of Service
RSVP	Resource reSerVation Protocol
RTT	Round Trip Time
SLA	Service Level Agreements
TCP	Transmission Control Protocol
TSW	Time Sliding Window
TTL	Time To Live

UDP	User Datagram Protocol
VPN	Virtual Private Networks
WAN	Wide Area Network
WDM	Wavelength Division Multiplexing
WWW	World Wide Web

Chapter 1

INTRODUCTION

1.1 Motivation for this Research

1.1.1 Current IP Networks and Real Time Applications

As we enter the 21st century, the Internet continues to experience tremendous growth. This growth can be seen in all aspects: the number of users, the amount of traffic, the increasing capacity of links and higher processing speeds of network equipment, the growth rates of Internet Service Provider (ISP) networks. A significant amount of traffic associated with various applications, such as WWW, FTP, Email, Telnet, video and voice, goes through the network, as this list of Internet-based applications and supporting tools continues its fast growth. In the near future, people will enjoy other multimedia products in distance learning, distributed simulation, distributed work groups and other areas. These new applications and services will change not only the computer and communications world, but also our daily life in a way never seen before.

However, the appearance of new applications has generated new requirements on the quality of service expectations we have of the Internet. In its earlier days, the Internet was developed to connect computing devices, exchanging and sharing data. Most of the data was of a textual nature, without any special quality of service requirements attached to

them. The existing best effort model of the Internet, which treats all traffic streams the same way, was able to accommodate these applications satisfactorily. However, the best effort model cannot provide the explicit quality of service guarantees the new applications need in terms of packet loss, end-to-end delay and delay jitter. In addition, several of these sources (such as, for example, variable bit rate video and Internet data) have bursty traffic behavior. Without suitable admission control policies, such traffic could generate severe congestion in the network, affecting the performance of many of the applications running through it.

Taking into consideration the points made above, it is evident that the existing “best effort” architecture is not adequate for the Internet to meet the new requirements placed upon it. Responding to this realization, the engineering and scientific research and development community specializing in telecommunications, networking and information technologies worked aggressively towards implementing the evolution of the Internet from its existing state to the global integrated services network of the future. Most of these activities were conducted under the auspices of the Internet Engineering Task Force (IETF) [1].

Providing Quality of Service (QoS) and traffic engineering capabilities in the Internet is essential, especially in supporting the requirements of real-time, as well as mission-critical applications. For this purpose, the current Internet must be enhanced with new technology that enables it to offer such capabilities for controlling its behavior as needed. Differentiated services and Multi-protocol Label Switching (MPLS) are emerging

technologies, which play a key role in IP networks by delivering QoS and traffic engineering features.

1.1.2 The Need for Network Performance Monitoring and Measurements

While there is no doubt that considerable progress has been already made in the area of QoS over the Internet, there is still a considerable amount of work to be done. One of the key questions remaining to be answered is how applications (especially those sensitive to delay and loss) behave when these new protocols and network architectures are used.

The reliability of broadband-integrated telecommunication networks relies on complex management functions that must be properly designed in order to allow the network to provide a wide range of services with different quality requirements. The global traffic offered to a broadband network is in fact becoming ever more heterogeneous, since new applications like digital TV, video on demand and cooperative working are now being requested, in addition to the classical bearer services. This complex mix has to be transferred in a reliable way; therefore, besides traffic modeling, which is already quite extensively considered in the networks, proper monitoring and performance evaluation techniques have to be developed in order to make network management effective and timely [2]. A test and measurement system is needed to monitor the status of packet networks at the end-to-end level, and evaluate the quality of service actually provided to the end users. While some applications are relatively insensitive to congestion, others are highly sensitive to packet losses, end-to-end delay and delay jitter. Classic examples are voice and video applications. The appearance of new devices and applications will only increase the need for the timely delivery of information. Distribution of real-time voice

traffic is basically different from traditional reliable data transfer, such as file transfer, remote login and electronic mail, since real-time voice data is sensitive to end-to-end delays and variations in the delays. Traditional data traffic requires a transmission channel with a low error rate and some minimum bandwidth. The performance metrics of these applications are typically average packet delay and throughput.

As described in [3], network measurements and monitoring are needed for network design, capacity planning and forecasting, operations and management, and customer-driven activities. The goals of measurements and monitoring are to:

- Determine the operational state of the network, including fault detection.
- Monitor the continuity and quality of network services, to ensure that QoS objectives are met for various classes of traffic, to verify the performance of delivered services, or to serve as a means of sectionalizing performance issues seen by a customer.
- Evaluate the effectiveness of traffic engineering policies.

There have been several research efforts on network performance measurements and analysis [4, 5, 6, 7, 8, 9]. All these efforts have reached the common understanding that the establishment of an Internet performance measurement infrastructure is essential for the optimal operation and healthy growth of the Internet.

1.2 Network Performance Measurement Metrics and Methods

1.2.1 Network Performance Metrics

Two approaches of the QoS parameters definition currently exist. ITU-T, in general Recommendation I.350 [10] or for IP in Recommendation I.380 [11], adopts a statistical probabilistic definition for the QoS parameters. The framework for performance measurement has been standardized [9]. Contrary to this, IETF documents are based on a deterministic philosophy. The principal document is Framework for IP Performance Metrics [8]. In the following sections we will look at the main ideas behind the defined performance metrics applied to the Internet environment.

1.2.1.1 Delay

The one-way delay metric, as defined in [12], is one of the basic quantitative characteristics of the network propagation delay. It is preferred to round-trip delay as there may be a significant asymmetry in paths to the destination and back to the source, and even if the two paths are symmetric, they may perform differently due to queuing.

Once a packet has been transmitted from the source, it travels through the network towards the destination and is processed by routers on the way. The metric is defined as the difference between the wire-time of the first bit of the packet at the transmitter, and the wire-time of the last bit at the receiver. The time it takes for the router to process some portion of the packet in order to determine the next hop is heavily dependent on the capacity of the router itself and the current traffic going through the router. A packet may be buffered before the router completes the processing of the preceding packets. Total time for this waiting process is called queuing delay. It is accumulative along the packet route.

The metric involves an upper bound of delay and says that the packet is considered lost and the value of the metric is undefined if the last bit does not arrive within that predefined period of time. Therefore the QoS performance of this flow is affected. It is particularly important to real-time applications, such as voice over IP, video conferencing, etc

One-way delay is usually measured through time-stamping a packet as it enters the network and comparing that timestamp with the time the packet is received at the destination. This assumes that the clocks at both ends are closely synchronized. For accurate synchronization, the clocks are often synchronized with the Global Positioning System (GPS).

1.2.1.2 Jitter

The variation in the delay of packets as they flow from one host to another is sometimes called “jitter.” It is unavoidable that packet delay variations (jitter) will occur within the network, due to the variability of the traffic when statistical multiplexing is used. Traffic variability generates variation in the queuing delay experienced by different packets. Packet delay variation is defined differently in IETF and ITU-T. As described in [13], the IP community therefore uses the term IP Packet Delay Variation (IPDV), defined, for a selected pair of packets in a stream of packets, as the difference in delay between two consecutive packets. Given a stream of delay measurements, IPDV is easy to extract. A two-point delay variation, as defined by ITU-T, is the difference between the one-way delay for each packet and the mean delay during the measurement period. Note that since it is a difference in delays between packet pairs, clocks do not need to be carefully synchronized.

1.2.1.3 Loss

Type-P One-way Packet Loss metric is defined in [14]. Packet losses can occur either due to buffer overflows, conflict, interference in transmission medium, or expiration of packets. If the packet does not reach the destination within a certain time after its generation and transmission, it becomes obsolete and will be dropped by the decoder (or if the network possesses “intelligence”, it could be dropped at a network element that detects expiration).

The measurement methodology relies on the one-way delay. Packet loss is a serious matter for real time applications, the degree of sensitivity depending on the application (e.g. voice, audio, video), the compression scheme used (e.g. H.263 or MPEG-2), and the specific application (e.g. small screen desktop video-conferencing or large screen, movie-quality video). In general, voice has a relatively lower sensitivity to packet loss rate (it can tolerate losses as high as 10^{-2} [15]), while video tends to be quite sensitive (packet loss ratio requirement below 10^{-4} is quite common in desktop video-conferencing; for applications requiring high quality visual information, it should be lower than that). Hence, maintaining the packet loss rate below a certain threshold is a key task in providing QoS to end user applications.

Currently, the Internet community primarily uses only one statistically derived quantity for loss, namely the loss average, defined as the average of the singleton loss values over a series of sent packets. The loss average corresponds to the loss rate, which is given as a percentage between 0 % and 100 %. There is also a discussion in the IP Performance Metrics Working Group [16] about ‘loss pattern sample metrics’, such as loss burstiness patterns, loss-free seconds, or conditional loss probability.

1.2.1.4 Throughput

Throughput is expressed in data units per period of time and is the long-term average utilization of network resources. ITU-T [10] currently defines IP packet throughput as the total number of successful IP packet transfer outcomes during a specified interval, divided by the time interval duration.

In general a client will desire a higher throughput. The throughput characteristics also provide an indication of the state of congestion in the network. A relatively better throughput at a certain time for a source means that the network is less busy at that time.

1.2.2 Network measurement approaches

The methods for measuring and monitoring network performance parameters usually fall into two categories: passive methods and active methods. A hybrid method is a combination of the traditional active and passive methods. The introduction of sampling into monitoring and control functions aims to make them less invasive and more efficient. Each method has its advantages and disadvantages. All the methods are discussed below.

1.2.2.1 Active Measurement

Active measurement generates controlled probe traffic and injects it into an ingress port of the network, and measures the received traffic at a receiving node. The method is becoming increasingly important due to its great flexibility, intrinsically end-to-end nature, and freedom from the need to access core network-switching elements. Thus active measurement methods are typically used to obtain end-to-end statistics such as latency, loss and route availability. In order to obtain measurement results that are representative for certain applications, active measurements require the generation of appropriate (artificial)

test traffic to emulate the expected traffic mix. Furthermore, test traffic always generates additional load on network links and routers, and can significantly influence the measurement results.

1.2.2.2 Passive Measurement

Passive measurements are based on the actual payload traffic in the network. Unlike active methods, passive methods do not add extra traffic load to the network. Besides this non-intrusive character, they provide a statement about the treatment of the current traffic in the observed network section. This is the case for most applications where statements about the actual situation in the network are required (like Service Level Agreements (SLA) validation, accounting, traffic engineering). Passive measurements are perfectly suited for the validation of SLA and are usually used to measure at-a-point metrics pertaining to certain network elements. Examples are link throughput and packet size statistics. However, recording and logging of packet traces in high-speed networks often require special gather, storage and processing of very large amount of data.

1.2.2.3 Hybrid Monitoring Method

The hybrid method introduced in [17] is a combination of the traditional active and passive methods in measurements; it can benefit from the advantages of both techniques, but also share some of their shortcomings.

Some of the advantages of the hybrid method are:

- The three main categories of performance parameters (packet delays, packet losses and throughput) can be obtained at the same time.

- The monitoring blocks (expressed in terms of time periods or number of packets) make it possible to describe the loss process and throughput in more detail than overall long-term metrics.
- Packet filters and traffic meters are flexible tools for carrying out the monitoring functions adapted to different QoS classes, protocols, port numbers and applications.

1.2.2.4 Sampling Methods

Sampling techniques are used to study the behavior of a population of elements based on a representative subset. Cochran [18] and Krishnaiah and Rao [19] introduce the basic concepts and process of sampling algorithms, and provide a detailed explanation of the statistical determination of the appropriate sample size for estimating a given parameter of a population, such as the mean value. In general, the samples are taken periodically at some fixed interval or in random distributions, in accordance with a given characteristic. In packet networks, performance parameters are computed by way of choosing some particular packets among those crossing the network. There are three conventional sampling methods (systematic, random and stratified sampling) employed by network management systems and measurements:

- Systematic sampling, or periodic sampling, deterministically samples data at a fixed time interval.
- Random sampling employs a random distribution function to determine when each sample should be taken. The distribution may be uniform, exponential,

Poisson, etc. Random sampling may take a varying number of samples in a given time interval.

- Stratified random sampling combines the fixed-time interval used in systematic sampling with random sampling, by taking a single sample at a random point during a given time interval.

In order to reduce the quantity of overhead resources required in processing the collected data and hence speed up all the further management functions, there is a fundamental design trade-off in selecting a network measurement and monitoring method between overhead resource costs, and the accuracy and timeliness of the data. In order to plan a suitable sampling strategy it is therefore crucial to determine what information is needed and the desired degree of accuracy in advance. Sampling-based measurement methods provide adequate techniques for reducing the quantity of control data and attract growing interests. A new working group on this topic PSAMP [20] was just formed in IETF, the Internet standards forum.

1.3 Previous Research Work and Problem Statement

Sampling has been applied to measurements for different purposes. Statistical sampling of network traffic was first used in [21] for measuring traffic on the NSFNET backbone in the early 1990s. Systematic, stratified and simple random sampling techniques are used to estimate the distribution of packet sizes and inter-arrival times from packet trace. Count-based sampling has already been implemented in products like Cisco's NetFlow [22]. In [23], the problem of identifying large flows is studied. A probabilistic packet sampling

method is used to identify large flows, and sampling probability is computed for each packet based on its size. This method requires each packet header to be inspected.

Traditionally, network management has made use of simple, non-adaptive sampling techniques. Such techniques use a fixed rule to determine when to sample data in each agent. The sampling rule can be deterministic, such as in periodic sampling, or it can involve a random component. A method for adopting a random sampling scheme to the required level of accuracy is shown in [24]. It introduces the sampling into monitoring and control functions to reduce the amount of resources (bandwidth, time, memory, processor time) required to deal with the data processing. That is obtained through a proper selection of the right information in order to make sampling-based monitoring and management functions less heavy and more efficient. It also addresses the evaluation of QoS parameters (loss and delay) by means of the application of the Simple Random Sampling technique to the simulation model of the measurement instrument. The main finding of this work is surely represented by the attained relationship between the accuracy (standard deviation of the estimated performance parameter) and the quantity of sampled data.

The deployment of sampling techniques for SLA validation allows for a reduction of the resource consumption. Instead of measuring all packets, only a subset of the observed packets is selected. With this subset an estimation of the metric of interest is calculated. The approaches for the deployment of sampling methods for non-intrusive delay two-point measurements are described in [25], a simulation study using a real trace to show that the validation of delay guarantees can be modeled as an estimation of packet proportions. Experiments are performed with systematic, random, and stratified sampling. It is shown

that the accuracy of estimation can be further improved if stratified sampling, which takes a-priori information into account, is used, rather than systematic sampling and random sampling. However, for SLA validation sometimes measurement results from different measurement points have to be correlated (e.g. for delay measurements). This requires the synchronization of the sampling processes at the involved measurement devices. Since passive measurements are based on methods where packets are neither marked nor modified in any way. If more than one measurement point is involved, the measurement results (timestamps and packet ID) from the different measurement points have to be collected at a common location in order to calculate the delay value. The possibilities for synchronization in the sampling process depend on the chosen sampling trigger, which is described in detail in Chapter 2. Due to reordering and loss of packets, the position of packets in the flow usually differs at measurement points in different places in the network. It is nearly impossible to synchronize two count-based or time-based sampling processes at different points in the network. Therefore the recognition has to be based on fields that already exist in the packet, which can be achieved by direct comparison of parts of the packet with a reference pattern [24] or by matching the results of a function performed on the packet content [26]. Furthermore, recording and logging of packet traces in high-speed networks often requires special arrangements for collection, storage and processing of very large amounts of data.

T.Lindh [27] presents a framework for embedded performance monitoring to measure QoS parameters in IP networks, especially virtual private networks. The proposed architecture is based on dedicated monitoring packets, which are inserted periodically between blocks of ordinary data packets, and experience the same conditions as the user

traffic. Measurement data carried by the monitoring packets may be processed to estimate losses, delays, delay variations and throughput. A combination of traffic meters and dedicated monitoring packets are used to obtain these goals. The delay measurements rely on a sampling procedure carried out by means of the monitoring packets, while packet losses and throughput are measured passively via traffic meters. The main purpose of embedded monitoring method is to measure and estimate losses, delays, delay variations and throughput, using a single procedure instead of several different methods. This method is based on an active technique that reflects the performance of the actual user traffic. The monitoring blocks (expressed in terms of time periods or number of packets) make it possible to describe the loss process and throughput in more detail than overall long-term metrics.

Based on the previous research work above, because of its simplicity, flexibility, and intrinsically end-to-end nature, the embedded monitoring method is the main test and measurement system under consideration. This performance measurement method is typically used to obtain end-to-end statistics, such as delay, jitter and loss. There are still some issues to be solved for achieving gains in precision or efficiency using the embedded monitoring method.

The performance of systematic sampling in the presence can be improved in several ways. One is to introduce a timer that sets an upper limit for the acceptable maximal time duration for a monitoring block, in case the time distance between two packets in a block can be quite long. Another one is to vary the block size so that the correlation with periodic traffic patterns could be avoided. One possibility is for example to let the block size vary

randomly. Finally, we find that none of these current sampling techniques for measuring the quality of service parameters (delay, loss, jitter, throughput) address the issue of how to carry out the sampling in an adaptive fashion, so that the accuracy is better if we know something about the traffic type (voice, video and Internet data) and traffic parameters (average rate, burst size, packet length distribution).

Since network traffic is frequently aperiodic, the rate for any of these non-adaptive sampling techniques is typically set based on the expected average network load [28], the traffic distribution, or simply to a value that will yield an acceptably small amount of overhead. If the actual traffic differs from the expected pattern, the measurements may prove to be inaccurate or may take an excessive number of samples.

To address these issues, we need to investigate how to combine time-based trigger and count-based trigger mechanisms efficiently, and then introduce the concept of adaptive sampling based on the estimated traffic rate to determine the number of monitoring packets inserted in a specific time interval and randomly insert the monitoring packets within the limited time period. Adaptive sampling techniques can be employed to dynamically adjust the sampling interval and optimize accuracy. Compared with the conventional sampling techniques, adaptive techniques can be used to adjust the time scale to insert the monitoring packets according to the availability (or lack) of information. By combining active measurement methods with the adaptive sampling technique, the injected monitoring packet patterns will depend on the current traffic statistics. The distance between monitoring packets will depend on the current traffic load. An increased block length during periods of heavy traffic can be adjusted to a desired average length. During a low

traffic rate, we can decrease the sampling rate to reduce the overhead traffic. For example, the Internet trace represents a random, bursty TCP/IP network data and includes traffic from HTML, FTP and Telnet activity. Adaptive sampling should be ideally suitable this type of data because the sample rate can be adjusted to compensate for the level of loading on the network at any given time.

We address issues related to the metrics, delay and packet losses, and how the network related parameters are “mapped” to the quality of the application. We believe that the results of this work are of importance to application developers and network architects. As will be seen in later chapters of this dissertation, our work has uncovered a number of mechanisms that have an impact on the performance of the application.

1.4 Objectives of the Thesis

The main focus of this thesis is the performance analysis of the proposed adaptive sampling to achieve gains in precision or efficiency. Based on the statistics of traffic coming from a video, voice or Internet data source, the adaptive sampling scheme is effective, provided that the appropriate sampling interval and rate can be identified and employed. Currently, providing Quality of Service and traffic engineering [1] capabilities in the Internet is essential, especially to support the requirements of real-time, as well as mission-critical applications. One important focus of the proposed research is to apply the new measurement methods to MPLS-based IP networks. Our research has three objectives:

- To evaluate the network performances based on the adaptive sampling measurements by using voice and video and Internet data applications over MPLS-based IP networks.

- To analyze how the adaptive sampling techniques achieve accuracy in network performance while maintaining the same sample counts or reduction of samples, and propose improvements if needed and if possible.
- To analyze the effect of the competing traffic on the adaptive sampling measurements over the complicated network topology.

The followings are the main tasks, which were carried out in this research:

- Study the sampling methods and develop the adaptive sampling approach and compare it to other monitoring methods.
- Consider the adaptive sampling method for use in the measurement system on MPLS networks.
- Develop simulation models based on a single MPLS domain and experiment with voice, video and Internet data sources.
- Analyze the results of the measurements and work out the improvement of the monitoring system.
- Make a proposal for enhancing the measurement methods and summarize the accomplished work.

1.5 Thesis Organization

The remainder of this thesis is organized as follows:

Chapter 2, “Adaptive Sampling Approach,” focuses on investigating the mechanisms that can be set up to adaptively adjust the parameters of the sampling technique based on the estimated traffic rate, and describes the features of the main parts of the adaptive sampling approach.

Chapter 3, “Simulation Models,” gives a brief introduction to voice, video and Internet data and describes the MPLS network that will be used in the simulation. This chapter also presents its implementation for evaluating different sampling approaches.

Chapter 4, “Analysis of Simulation Results,” introduces the evaluation criteria and evaluates the proposed adaptive monitoring method against other sampling methods for different traffic source types, and works out improvements to the monitoring method.

Chapter 5, “Conclusions,” contains a summary of key findings and some suggestions for further improvements and future research directions.

1.6 Thesis Contributions

The first contribution is that this research [29] develops the adaptive sampling schemes for monitoring and measuring network performance metrics in MPLS-based IP networks supporting voice, video and real-time data applications. Our adaptive sampling technique differs from the existing sampling techniques for network performance measurements in that it can eliminate the bias due to synchronization by combining count-based and time-based sampling to randomly inject the monitoring packets during the limited time period. The variance of the estimated values is reduced by employing Time Sliding Window (TSW) rate estimator for traffic rate estimation over time. We then investigate the role of

sampling in relation to each metric, and whether source statistics would affect the parameters of the sampling schemes. Adaptive sampling dynamically adjusts the sampling rate based on the observed sampled data. Thus, the mechanism can be set up to adaptively adjust the sampling technique in terms of the type of traffic, and it is able to increase the accuracy of measurement and reduce the load on the network and the network manager.

The second contribution is to design and develop a simulation model that is flexible for assessing sampling measurement methods for different traffic models and network topologies.

The third contribution is network performance evaluation through different sampling techniques using three types of traffic: 1) voice, 2) video and 3) Internet data by end-to-end performance evaluations, and also evaluation of the factors (the buffer size and utilizations) impacting on the performance of the network model through different sampling techniques. Accuracy of the sampling / estimation procedure for delay and delay jitter is assessed with the confidence interval around estimated values. The simulation results show the adaptive technique performs well on independent or short-term correlation traffic such as voice and video. This approach is able to achieve the desired accuracy of the average delay and delay variation and packet losses. Moreover, this approach has a number of desirable properties. Firstly, simple processing for the monitoring packets, they are directly observed and give direct measurements of end-to-end path characteristics, such as delay and packet loss rate. Its advantage is that it does not require any measurement support from the network. Secondly, with the competing traffic increasing, the method provides acceptable estimates of the network performance parameters. The competing traffic has less effect on the

accuracy of this approach to measure end-to-end path characteristics for MPLS-based IP networks.

The forth contribution is to achieve an understanding when adaptive sampling is advantageous.

Chapter 2

ADAPTIVE SAMPLING

APPROACH

The adaptive sampling approach for network performance measurements is proposed to reduce the number of monitoring packets that are required to achieve a desired level of accuracy, or conversely, improve the estimation accuracy without increasing the sample size. For this we investigate the application of different sampling disciplines and the role of sampling in relation to each metric. We then introduce the concept of adaptive sampling with active measurement, and investigate how adaptive techniques can be used to adjust the sampling rate for each parameter monitored according to the availability of information about source statistics. Furthermore, adaptive schemes with a dynamic adjustment of the generation of monitoring packets are developed. The improvement in accuracy is achieved by incorporating a-priori information about traffic characteristics into the sampling process.

2.1 Classification of Sampling Methods

The classification of sampling methods is based mainly on the sampling algorithm, the trigger used for determining a sampling frequency and a sampling interval (the period of

time or the number of elements). These parameters and their potential values are described in detail.

2.1.1 Sampling Algorithm

A sampling algorithm describes the basic process for the selection of samples. In an effort to balance accuracy and sampling overhead, according to previous research work [17, 20, 21, 22, 23, 24, 25], several sampling disciplines have been applied to network managers and network performance measurements.

2.1.1.1 Conventional Sampling

Traditionally, simple, static sampling with a fixed sampling rate is used for network management and network performance measurement. Such techniques make use of a fixed rule to determine the sampling process and functions. The sampling processes can be deterministic, such as systematic sampling, or random. There are three main classes of conventional sampling schemes used in network management systems:

Systematic sampling

Systematic sampling describes the process of selecting the sampling intervals from the population, according to a deterministic function. This can be obtained by periodically selecting every n th element of a trace, or selecting all packets that arrive at pre-defined points in time. This method is particularly attractive because of its simplicity. It eliminates the need to generate a random number before the next new packet is selected. Its disadvantage is its vulnerability to sampling bias. Related to this problem is that systematic sampling can easily be anticipated [30]. If the metric being measured itself exhibits

periodic behavior, then there is a high risk that the estimation will be biased for reasons of synchronization. Figure 2.1 illustrates the case least favorable to systematic sampling. This case presents that every observation within the systematic sample is exactly the same, so that the sample is no more precise than a single observation taken at random from the population. As shown in Figure 2.2a, systematic sampling can take a sample with a fixed time interval of T seconds.

Figure 2.1 Systematic sampling

Random Sampling

Random sampling selects the sampling intervals in accordance with a random process. The selection of elements is independent and each element is chosen with an equal probability. Significant drawbacks to the process include the fact that the random sampling selection is not truly straightforward, and requires the generation of the random numbers.

For simple random sampling, the estimation accuracy can be calculated from the variance of the characteristic of interest in the parent population and the number of elements in the sample.

$$\sigma = \sqrt{s^2 \frac{1-f}{n}} \quad (2.1)$$

Here, σ is the standard deviation for simple random sampling, s^2 is the population variance, n is the number of elements in the sample, and f is the ratio of the number of sample n to the population size.

Therefore a given accuracy level (expressed as confidence interval boundaries and level of confidence) still requires a minimum amount of packets to be sampled. Sampling fewer packets would lead to a decrease in the accuracy of the estimation. Figure 2.2b shows random sampling selecting a varying number of samples in a period of T seconds.

Stratified Sampling

Stratified sampling divides the sampling process into multiple steps by using a-priori information, which is used to perform an intelligent grouping of the elements of the parent population. The subpopulations are called strata. A simple random sample is then taken from each subset. The stronger the correlation between the characteristics used to group the population and the investigated characteristics (which we are interested in measuring), the higher the accuracy of the estimation, which can be achieved. For instance, if the dividing characteristic were equal to the investigated characteristic, each element of the sub-group would be a perfect representative of the overall population. In this case it would be

sufficient to take one arbitrary element out from each subgroup to get the actual distribution of the characteristic in the parent population. Although the values of the investigated characteristic cannot be accessed in practice, the stratification variables (the packet size or the time interval) are known for all elements of the parent population. Therefore large gains in precision are generated if the population is composed of subgroups, which vary widely in size, or the principal metrics to be measured are closely related to the sizes of the subgroups.

The idea in [20] introduces an example of stratified sampling where the trace is divided into time intervals (subgroups) based on a time-based systematic sampling. One packet per interval is then selected, following a random sampling process. Figure 2.2c illustrates stratified random sampling in a given time interval.

In [31], Tanja assumes a correlation between the packet size and the delay experienced by the packet, and shows how stratified sampling based on packet sizes can be used to significantly reduce the sampled packets, while achieving a given accuracy level of packet delay. Tanja investigates different stratification strategies with different allocation schemes, different number of strata and different methods for calculating the boundaries.

In our work, we investigate stratified sampling where the trace is divided into subgroups based on count-based systematic sampling. One packet per interval is then selected, following a random sampling process.

Figure 2.2 Conventional sampling methods

2.1.2 Sampling Frequency and Interval Length

According to [20] and [32], sampling techniques can use packet counts or timers to trigger the selection of a packet for inclusion in a sample, because the unknown relationship between the number of packets and the duration of an interval can lead to unexpected overlapping of sampling intervals. The trigger determines what kind of event starts and stops the sampling intervals. With this sampling frequency, the length of the sampling interval can be determined.

Some common triggering mechanisms are described below.

Count-based Trigger

Count-based mechanisms trigger the start and stop of a sampling interval in accordance with the number of the elements. One example is systematic sampling, which deterministically selects every n th element (packet) of the data trace. The measurement of the delay requires that each packet be analyzed and captures the same packets between different measurement points. For active measurement, the monitoring probes can be generated according to the number of packets counted. It is nearly impossible to synchronize two count-based sampling processes at different points in the network for one-

way delay measurement using passive measurement. Since the reordering and loss of packets occur, the position of packets in the flow usually differs at measurement points in different places in the network.

Time-based Trigger

Time-based sampling methods use a timer (a specific time period) rather than a packet counter to trigger the selection of packets to be included in the sample. When the timer expires, the next packet to be captured or generated is selected. Since the number of packets arriving in a specific time interval is unknown (and may be zero), the requirement of a minimum sampling size has to be taken into account.

Time-based sampling is particularly bad for assessing inter-arrival times, since one tends to miss bursty periods, which contain many packets of relatively small inter-arrival times. For the remainder of discussion, therefore, we will focus on only two of the count-based methods: systematic and stratified random sampling.

However, with this method it is impossible to ensure that all sampled packets will be captured at different measurement points for one-way delay computation using passive measurement. As for count-based sampling, the capturing of the full elements at one measurement point is one option to avoid that problem.

Packet-content- based Trigger

With this method the content of the packet itself triggers the sampling process. The selection process is based on information contained in the packets and therefore it is

guaranteed that the same packets are captured at both points. Thus, a packet-content-based selection is suitable for passive measurements at multiple points. On the other hand, this technique incurs higher resource consumption than time-based and count-based sampling, since the post-processing of every packet is necessary in order to find whether the packet belongs to the sample or not, and time-based and count-based sampling simply depend on the arrival times or counters for the packet events.

2.1.3 Limitations of Conventional Sampling

By investigating the conventional sampling methods, the limitations of the conventional sampling are presented.

- Systematic sampling problems

Systematic sampling is based on deterministic functions, and the sampling decision for a packet can either use time-based or count-based sampling methods. Time-based sampling is particularly bad for assessing the network performance metrics such as delay and delay variation, since one tends to miss bursty periods, which contain many packets with relatively small inter-arrival times if using the larger timer, and to inject unnecessary monitoring packets during silent periods, which contain less packets if using the smaller timer. Using count-based sampling can avoid the problems introduced by time-based sampling, but there is a high risk that the estimation will be biased for reasons of synchronization if the metric being measured exhibits periodic behavior. If the monitoring packets with timestamps are sent randomly during limited periods it is possible to estimate the delay and delay variation more accurately.

- Stratified sampling problems

Stratified sampling divides the trace into subgroups based on count-based or time-based systematic sampling. A simple random sample is then selected from each subgroup to get the actual distribution of the characteristic in the parent population. Therefore, stratification may produce a gain in precision in the estimates of characteristics of the whole population than systematic sampling, but the variance of the estimated value may be still large if the population characteristic is independent or short-term correlation such as voice traffic. The mean value can be estimated by the weighted sum of the mean values of the subgroups. The variance of the estimate for a stratified sampling with proportional allocation of elements that have to be selected per subgroup is calculated as follows.

$$s^2 = \frac{1}{n} \sum_{h=1}^L \frac{N_h}{N} \cdot S_h^2 \quad (2.2)$$

Here, N is the number of population, n is the number of the elements in the sample, L is the number of the subgroups, S_h^2 denotes the variance of the investigated characteristic (delay) within the subgroup, and N_h denotes the number of elements in the subgroup. If we reduce the variance of the investigated characteristic within the subgroup, and then the variance of the estimate for the stratified sampling can be reduced.

Based on the problems and the analysis of the two conventional sampling methods, we propose the adaptive sampling method depending on the traffic rate. One advantage of this method is that it can eliminate the bias due to synchronization by combining count-based and time-based sampling to randomly inject the monitoring packets during the limited time

period. Another advantage is to reduce the variance of the estimated values by employing TSW rate estimator for traffic rate estimation over time. The rate estimator accommodates both the burstiness and the silence often observed in the traffic, and smoothes out its estimate to approximate the actual sending rate. Since the number of monitoring packets over the fixed time interval is adaptively adjusted with the estimated traffic rate, the variance of the investigated characteristic (delay) within the time interval may be reduced. Thus, the variance of the estimate values for the adaptive sampling can be reduced.

2.2 Adaptive Measurement Mechanism

The basic idea of the proposed measurement method in this study is to use the adaptive sampling technique in conjunction with the monitoring packets, which are inserted into the traffic. First, we need to solve how adaptive sampling can be used to optimize the sampling rates dependent on the network traffic characteristics, and then apply adaptive sampling with the embedded monitoring method to measure network performance parameters.

2.2.1 Adaptive Sampling Technique

A key element in adaptive sampling is the prediction of future behavior based on past-observed behavior. Since network traffic characteristics vary significantly over time and the characteristics are not known before sampling rates or stratification are selected, conventional sampling methods with pre-defined sampling intervals introduce unnecessary over-sampling or under-sampling during the periods of measurement. It is likely that the optimal sampling rates for one segment of the trace are not suitable for other segments. The best method would be to adapt the sampling rates to the characteristics of the measurement interval itself; however, the current characteristics cannot be accessed. That means we use

the characteristics of the previous block to estimate the characteristics of the current block, and derive from those the sampling rates for the next block. If the prediction is accurate, the sampling rate can be reduced. Inaccurate predictions indicate a change in network behavior and require an increased sampling rate to determine the new pattern.

Time Sliding Window (TSW), a probabilistic tagging algorithm for marking packets, is proposed as one part of DiffServ mechanisms in [33] [34]. The design of TSW is very simple. The TSW tagging algorithm consists of two independent components: a rate estimator that estimates the sending rate over a certain period of time, and a tagger that marks packets as IN or OUT based on the rate reported by the rate estimator. The rate estimator accommodates both the burstiness and the silence often observed in TCP traffic, and smoothes out its estimate to approximate the actual sending rate. With the estimated rate and the target rate, if the sending rate exceeds the target rate, the tagger will mark those packets as OUT, and conversely, the tagger will mark those packets as IN.

Based on the principle of the TSW algorithm, the TSW rate estimator for traffic rate estimation over time is employed in this research. The sample size (the number of the monitoring packets inserted into the traffic) for the current time interval can be adaptively adjusted based on the rate of the previous time interval reported by the rate estimator. The adaptive sampling mechanism has three distinct parts: a rate estimator, sample size estimation algorithm and sampling scheme. The logical relationship of the three parts is shown in Figure 2.3 below.

Figure 2.3 Procedure of adaptive sampling

2.2.1.1 Time Sliding Window Rate Estimator

The traffic estimator is used to estimate the traffic rate. The estimated value determines the capacity demand from a certain edge node to a certain destination node. The principle of TSW is extremely simple. TSW maintains only three state variables:

Time_Win: is measured in specific units of time;

Avg_rate: the rate estimate upon each packet arrival;

T_{front} : is the time of last packet arrival.

TSW is used to estimate the rate upon each packet arrival; thus, variables Avg_rate and T_{front} are updated each time a packet arrives. A description of the main steps of the TSW algorithm follows.

Initialization:

$Time_Win = a\ constant;$

$Avg_rate = 0;$

$T_{front} = 0;$

Upon each packet arrival, TSW updates its state variables as follows:

$Bytes_in_TSW = Avg_rate * Time_Win;$

$New_bytes = Bytes_in_TSW + pkt_size;$

$Avg_rate = New_bytes / (now - T_{front} + Time_Win);$

$T_{front} = now;$

Figure 2.4 illustrates how TSW calculates Avg_rate . Three packets arrive at time $t1$, $t2$ and $t3$ respectively. The lightly shaded box depicts TSW rate estimator. The width of the box is the $Time_Win$, along the Time axis. The height of the box is Avg_rate ; the higher the box, the higher the estimated Avg_rate . The right edge of the box is T_{front} , or the time of last packet arrival. The area of the shaded box indicates the bytes contained in TSW, or the amount of memory that TSW retains concerning a particular flow.

TSW works as follows:

- At time $t1$, TSW calculates a new *Avg_rate* after a packet arrival. The height of the shaded box is the newly recorded *Avg_rate*.
- At time $t2$, a new packet (indicated in dark) arrives. The box area depicts the size of the new packet in bytes.
- Both the bytes in its memory and the bytes in the new packet spread across the sum of its *Time_Win* and inter-packet arrival time. This is depicted in Figure 2.4, where both the shaded box and the dark box are spread across the time span of $(Time_Win + (t2-t1))$.
- The decaying phase of TSW remembers only the *Avg_rate* of its newly calculated rate, and forgets everything beyond a *Time_Win* period of time. This happens when TSW resets its *T_front* to be the current time.
- A new packet arrives at time $t3$. TSW restarts its cycle of incorporating the new arrival rate and decaying the past history.

Figure 2.4 TSW rate estimator

The variable *Time_win* depends on two factors. On one hand, the *Time_win* should be much bigger than the typical Round Trip Time (RTT) of an IP flow, since TSW keeps a long-term perspective on estimating sending rates. On the other hand, a value of *Time_win*

should be small enough so that the rate estimator is sensitive to the changing rate of the connection. In our simulation experiments, we use *Time_win* to be between 0.05 seconds and 0.1 second.

2.2.1.2 Sample Size Estimation Algorithm

Measurement time is divided into (non-overlapping) observation periods (referred to as time blocks). The sample size (the number of monitoring packets inserted into the traffic) for the current time interval can be adaptively adjusted based on the rate of the previous time interval reported by the rate estimator.

The number of monitoring packets over the fixed time interval is adaptively adjusted with the estimated traffic rate. Upon the arrival of each fixed time interval, the sample size is updated as follows:

$$Sample_size = Avg_rate * time_interval / (block_size * pkt_size)$$

Whereas *pkt_size* is the average packet size, *block_size* is the number of data packets between the monitoring packets. This parameter can be chosen based on the observed population values (the estimated traffic rate in this research). The *Sample_size* is the number of monitoring packets based on *Avg_rate* and *block_size* during the appropriate chosen *time_interval*.

2.2.1.3 Embedded Monitoring Packets

In accordance with the comparison of passive and active measurement described in Chapter 1, active measurement is the test and measurement system considered in this study. The purpose of embedded monitoring is to measure relevant network performance

parameters based on the actual user traffic. These dedicated monitoring packets are inserted between blocks of ordinary data packets, as shown in Figure 2.5. The sending node generates monitoring packets that convey monitoring information between every N user packets on average. The receiving node detects, processes and analyzes the monitoring packets.

The main performance parameters (packet delays, delay variations, packet loss ratio) can be obtained at the same time through this method. The structure of the monitoring packet is depicted in Figure 2.5.

Figure 2.5 Format of the monitoring packet

We employ and improve the embedded monitoring method by adaptively generating the monitoring packets instead of periodically generating the packets into the user traffic, and insert the number of packets sent and a timestamp into the monitoring packets. The receiving monitoring systems detect the monitoring packets through the unique protocol

number in the header, and keep track of the number of received packets from the entry node, then process the collected data. For the next step, we need to consider how to allocate and insert these monitoring packets into the original traffic stream.

Based on the sample size calculated in a fixed time interval, the time interval is divided into the timers based on a time-based systematic sampling. One monitoring packet is then randomly inserted per timer, following a uniform distribution.

$$Timer = time_interval / Sample_size$$

The procedure is depicted in Figure 2.6.

Figure 2.6 Adaptive random sampling

2.2.2 Measurement Results

The monitoring functions are activated if certain sets of conditions that express the operator's monitoring policy are fulfilled; for example, if it is specified that the service

class should be monitored during certain time periods. These functions are implemented in the edge routers. The measurement data carried by the monitoring packets may be processed by the receiving system in an exit node. For examples, a sample of the transfer delay between a sending and a receiving node is given by the difference between the time at receiver and TimeAtSender in a monitoring packet. Lost monitoring packets are detected through missing sequence numbers.

2.2.2.1 Delay and Delay Variation

One-way delay is computed as the difference of the timestamps assigned in the monitoring packet header between the source and the receiver, provided that the clocks are synchronized.

The delay variation is then defined as the difference between the one-way delay for each monitoring packet and the mean delay during the measurement period [11] in the monitoring system. Since the timestamps are samples in the stream, the results depend on the sampling rate.

2.2.2.2 Loss

Today packet losses are normally presented as the long-term ratio between the number of lost packets and the number of transmitted packets. The loss ratio is the fraction of packets lost due to buffer overflow in the queue of the network. Since the number of packets transmitted (and received) and timestamps for every monitoring packet are recorded at every edge node, as a function of time, it is possible to obtain an estimate of the packet loss ratio by the monitoring packets. In fact, it is a cumulative quantity not affected

by a reduction of the data from which it is computed. Hence, sampling provides a definitely accurate method of monitoring on-line data loss.

Chapter 4 shows the results of our activity in both of these stages: studies of suitable sampling techniques usable in the monitoring system and the related estimation procedures running in the monitoring points, using different sampling techniques.

Chapter 3

SIMULATION MODELS

“Simulation is the imitation of the operation of the real-world process or system over time.” [35]. Simulation leads to improvements in cost reduction and shortens the research and development cycle with the ability to incorporate arbitrary levels of details. Simulation modeling is desirable when introducing new network performance measurement method such as the adaptive sampling technique or the network communication protocols. The simulation model is used to quantify the network performance and verify the functionality of the network. The first objective of modeling is to demonstrate that the MPLS-based IP network and its components can work correctly. As mentioned earlier, the use of realistic traffic models in the performance measurements of MPLS-based IP networks is absolutely essential for the accuracy of results and a successful design of new measurement technologies.

3.1 System Model

The main components of the system model consist of: traffic source model, MPLS network model, and the performance measurement meters. The structure of the system model is depicted in Figure 3.1.

Figure 3.1 Structure of system model

The phenomenal growth and popularity of the Internet leads to the expectation that in the future, significant amount of the traffic going through networks will be related to the Internet applications, such as, WWW, FTP, voice over IP and videoconferencing. Traditionally it was sufficient to model voice traffic for the telephone network and data traffic for computer networks. With the emergence of the Internet there will be a mixture of voice, data, video and still image the traffic on one network, see [36] for more details.

We have paid particular attention to real time application traffic models to obtain accurate results of increasingly popular multimedia data over these networks. These models (voice, video and Internet data) are based on either our measurements or the findings of the previous works. First, voice traffic produced by a slow speech activity detector, is modeled by a two-state Markovian model. Second, for video traffic we have used actual H.263 video streams. Finally, we will introduce Internet data traffic model that can capture long-range dependence characteristics of the aggregated traffic. Section 3.2 provides more details of the traffic source models.

Multiprotocol Label Switching (MPLS) is assumed because its packet forwarding paradigm and traffic engineering capabilities are seen as essential to providing end-to-end

QoS in IP networks [37]. An overview and main applications of MPLS are well described in [38] and [39] respectively. MPLS is an industry-standard approach developed by IETF for bringing the connection-oriented forwarding properties of layer 2 switching technologies while maintaining desirable flexibility and scalability of layer 3 routing [10]. By combining the best of network-layer routing and link-layer switching, MPLS extends the Internet routing with respect to packet forwarding and path controlling. In addition, MPLS uses label-swapping forwarding paradigm and speeds up packets forwarding to their destination in a more efficient way by eliminating the need for routers to perform an address lookup for every packet. The different network scenarios will be studied and provided in Section 3.3.

The performance measurement meter within the Ingress Label Edge Router (LER) consists of a packet classifier, counting and adaptive sampling functions, a process that sends (receives) the monitoring packets and facilitates the storage and processing of monitoring data. A packet classifier selects the relevant packets to be monitored from the aggregate user traffic incoming to the edge routers. The counters are used to keep track of the cumulative number of sent (received) packets. As explained in Chapter 2, the TSW algorithm and estimation sample size algorithm estimate the traffic rate and the sample size during the predefined time interval. The monitoring packets are generated and inserted based on the sampling scheme. These dedicated packets carry timestamps, the number of sent and a sequence number. The performance measurement meter within the egress edge router detects and captures the monitoring packets and computes the packet losses and delays between two edge routers. Details of the performance measurement meter component are given in Section 3.4.

3.2 Traffic Source Models

The traffic source models that we use in our simulation are: 1) voice, 2) video, and 3) Internet data. Each traffic source has its own characteristics related to the nature of its application category.

3.2.1 Voice Source Model

Voice signals are composed of periods with voice activity (talkspurt) and silence periods (gaps). These periods correspond to the talking, pausing, and listening pattern of conversations. The packet stream from a single voice source is characterized by the packet generations at fixed intervals of T ms during the ON state (talkspurt) and no packets during the OFF state (silence) as in Figure 3.2.

Figure 3.2. Voice packet generation pattern

To better utilize network resources, user should halt transmission during silence periods where there is no signal to be carried through. The voice coder of the terminal samples and digitizes the voice signal during talkspurt with sampling rate 8,000 samples per second. The sample is digitized. On the other hand, during the silence periods, it suppresses the signal. Two-state Markovian models described in [40] can be used to model the traffic

generated by a user equipped with slow speech activity detector. All spurts and gaps have exponentially distributed durations. The mean value of the talkspurt (ON) period is T_1 and the mean value of silence period is T_2 . This is shown in the following figure.

Figure 3.3 On-off Voice Source Model

As shown in the figure, the probability of transition from principal silent to principal talking with the elapsed time τ is

$$\sigma(\tau) = 1 - e^{-\frac{\tau}{T_1}} \quad (3.1)$$

The probability of the user remaining silent is then given by

$$\alpha(\tau) = 1 - \sigma(\tau) = e^{-\frac{\tau}{T_1}} \quad (3.2)$$

Similarly, the probability of transition from principal talking to principal silent is

$$\gamma(\tau) = 1 - e^{-\frac{\tau}{T_2}} \quad (3.3)$$

The probability of the user continue talking is

$$\beta(\tau) = 1 - \gamma(\tau) = e^{-\frac{\tau}{T_2}} \quad (3.4)$$

To specify the voice model completely, in our case, the packetization period is chosen as $T=16$ ms, the durations of both talkspurts and gaps are all with mean values of 1.0 second and 1.35 seconds respectively [41]. The voice packet size depends on the coding scheme. For instance, for 64Kbps PCM coding and $T=16$ ms, the packet size is 128bytes.

3.2.2 Video Source Model

A video stream is generated by sampling images (frames) of a scene at a certain sampling frequency. A high enough sampling frequency, typically between 20 and 30 images per second, will make the playback (when is played at this speed) to appear as a continuous motion picture. Each frame consists of a grid of pixels (picture element), each having a discrete value, or a set of discrete values, giving a measure for the intensity or color of the small square it represents. For grayscale images, the pixel value is typically represented using eight bits (one byte), giving possible values between 0 and 255 inclusively. The value usually represents the amount of light within the pixel: 0 is black and 255 are white while the values in-between give the various shades of gray. For color images, pixels normally consist of three bytes, describing a color in a certain color model. For example, the three values may represent the red, green and blue color components.

A video stream tends to be quite demanding when it comes to storage requirements and network resource requirements for electronic transfer. It is thus necessary to compress the data before being stored or transferred. Compression algorithms take advantage of similarities between nearby frames, since usually there are small changes from frame to

frame within a video sequence. To take advantage of the temporal redundancy, the pixel values in a block may be predicted based on blocks in nearby frames. When such prediction is used, the block is represented not by the actual pixel values, but rather by the differences from the matching pixel values in the frame used for prediction. To make prediction better, motion compensation is often used. A displacement vector may be associated with a block, describing how the block has moved relatively to the frame used for prediction. There are two video compression standards exist in ITU-T recommendations for video conferencing: H.261 [42] and H.263 [43].

H.261: This standard specifies a video encoding and decoding schemes for videophone, videoconferencing, and other audio-visual services. This recommendation is designed for ISDN-lines or other media and is based on transfer rates that are multiples of 64 Kbps. Frames are partitioned in blocks of 8×8 pixels, each of which are transformed, quantized and Huffman-coded separately. A macro block is defined as four neighboring luminance blocks, and one block from each of the chrominance components, making up a 16×16 sub-image. Two types of frames are defined; intra-coded frames and inter-coded frames. Intra-coded frames are coded as stand-alone frames, while inter-coded frames use prediction errors with respect to the previous frame. The sender may decide not to send blocks that haven't changed since the previous frame.

H.263: This standard works much like H.261, but there are several extensions, and some modifications. Extensions to H.261 include “PB-frames mode”, where two frames are coded as one unit. The latter frame is coded as an intra-frame, while the former frame is

coded in inter mode, possibly using bi-directional prediction between the previously seen frame, and the intra-coded frame of the same unit.

In our simulations, we use real H.263 video stream with 20 frames per second, captured by the network analyzer.

3.2.3 Internet Data Source Model

It is well known that highly aggregated Ethernet, variable-bit-rate video, FTP, and WWW traffics demonstrate self-similar behavior. Self-similarity manifests itself in several aspects. One is the implication of long-range dependence. This means that traffic intensity remains correlated with values in a distant past. Another feature of a self-similar process is the presence of burstiness at all time scales. A lot of traffic models have been proposed, which can reflect the statistical behavior of real network traffic. Some of the models are described in the papers by Tsybakov and Georganas [44], by Willinger [45], by Norros [46], and by José R. Gallardo [47].

In our simulation model, we use the alpha-stable traffic model, which was proposed and implemented by José R. Gallardo (see [13]). This traffic model not only reflects the self-similarity and long-range dependence characteristics of aggregated traffic, but also captures the appropriate level of burstiness of different types of traffic. The author has proved that aggregate traffic traces that present very high burstiness can be modeled accurately with self-similar processes by using alpha-stable distributions. According to [13], each traffic process has to be modeled as a function of a fractional stable noise process Y_f . The traffic model was defined as follows:

$$W_j = m + a \cdot Y_j \quad (3.5)$$

Here, W_j represents the actual number of bytes arriving during the j -th time interval of unit length. The parameter m is the mean value of the number of arrivals per unit time, a is a scale factor, Y_j is a Fractional Stable Noise (FSN) process with index of stability α and Hurst parameter H .

A slightly modified version of the previous model was proposed for some traffic exhibiting noticeable positive spikes, whereas the negative spikes are less pronounced (such as Ethernet and WWW traffic):

$$W_j = m + a \cdot (|Y_j| - \mu_Y) \quad (3.6)$$

Where μ_Y is the mean value of $|Y_j|$, m , a and Y_j have the same meaning as in Equation (3.5).

The traffic self-similarity coefficient H , as well as the parameters of the stable distribution, is estimated through analyzing different types of self-similar traffic traces. Figure 3.3 (quoted from [9]) shows an example of a traffic profile generated by the alpha-stable traffic model with the Hurst parameter $H = 0.834$, where α is equal to 1.95.

Figure 3.4 Self-Similar traffic generated by α -stable traffic model

3.3 MPLS-Based IP Network Model

Two relatively simple but representative network topologies based on an MPLS-based IP network are used in our simulations. The first one is the simple network topology used to verify the operation of the adaptive sampling method by comparing it to the conventional sampling methods. The competing traffic streams are used to demonstrate the effects of competing traffic on the performance of the sampling methods.

3.3.1 Experimental Simple Network Model

The basic operation of an MPLS network is illustrated in Figure 3.5, and also almost the same topology is implemented with OPNET 8.0 simulation tool for our experimental analysis of adaptive sampling techniques in Chapter 4.

At the ingress to an MPLS domain, a label-switching router classifies and routes IP packets into Forwarding Equivalence Classes (FECs), which is a group of packets that require equivalent forwarding treatment across the same path, based on a combination of information carried in the IP header of the packets and the local routing information maintained by the Label Switch Routers (LSRs). An MPLS header is then assigned for each IP packet to specify the path and its priority according to their forwarding equivalence classes. Each MPLS label stack header (also known as MPLS shim header) is 32 bits long and contains a 20-bit label, a 3-bit experimental field used for experimentation to indicate packet's treatments, such as queuing and scheduling, a 1-bit label stack indicator and an 8-bit Time To Live (TTL) field.

Within an MPLS-capable domain, each LSR examines the label and the experimental field and make packet-forwarding decisions by using this information as the index to look up the forwarding table of the LSR. The packet is then processed as specified by the forwarding table entry. The outgoing label may replace the incoming label, and the packet may be switched to the next LSR. Before a packet leaves a MPLS domain, its MPLS header is removed. This whole process is showed in Figure 3.5. The paths along which MPLS packets traverse are called Label Switched Paths (LSPs). MPLS uses some signaling protocols like RSVP, LDP or CR-LDP to set up LSPs. In our simulation model, the static LSP is used. The edge routers are connected via a logical LSP. The LSP is in congestion as its capacity is less than the sum of the bandwidths required by all the sources of the LSP. Each end host group (gateway) consists of multiple users. Thus, the traffic traversing the simulation network is an aggregation of multiple individual flows. The goal is to measure and estimate the packet loss, delay and delay variation for the aggregate IP traffic generated by the above source models on the LSP between edge nodes.

Figure 3.5 MPLS simple networks

3.3.2 Experimental Network Model with Competing Traffic

Measurements on non-traffic-engineered LSPs may be collected to support the possible future traffic engineering of those LSPs. In order to control the path of LSP effectively, we need to monitor and measure the network performance parameters accurately. The network model shown in Figure 3.6 is designed to evaluate the performance of the adaptive sampling method when there is the competing traffic on the observed path. As IP packets with different destinations enter the MPLS domain from multiple sources, the edge LSR 1 typically determines which LSP to use for each packet, and attaches a label to the outgoing packet accordingly. It is important to note that it is the label value at the ingress router that actually determines which LSP to travel for the packet. LSR 1 then forwards the packet via the appropriate interface for the selected LSP. As the intermediate LSR 2 receives each packet, it decides based on the incoming interface and label value, the outgoing interface and label value with which to forward the packet to the next hop. So, according to this diagram, at the ingress router, IP packets from source to destination LSR 2 and LSR 3 are mapped into the common label switched path 1 (LSP 1) by attaching label 21. At the same time, the competing traffic enters the MPLS domain at LSR 4 and packets are mapped into label switched path 2 (LSP 2) by attaching label 25, and merge the monitored traffic at the . As the packets traverse LSPs, the intermediate LSR 2 examines the label in the MPLS header of each packet, and looks up its MPLS forwarding table to match for the incoming labels. LSR 2 then swaps label 21 and the labels of the competing traffic with an outgoing label 15 before it forwards the packet to the egress LSR 3. Finally, at the egress LSR 3, all labels are removed, and packets are then forwarded using just the traditional IP forwarding paradigm to their respective destinations.

Choosing different volumes of the competing traffic, the LSP 1 may become congested as it competes with the competing traffic for enough bandwidth to carry its traffic. We can investigate the LSP1 measurement entities under impact by the competing traffic and compare the performance of the adaptive sampling with the conventional sampling methods applied for different traffic.

Figure 3.6 MPLS with competing traffic

3.4 Performance Measurement Meter

In this section we describe an implementation of the performance measurement meters where these functions constitute an integral part of a router; Figure 3.7 shows the state diagram of the monitoring procedure in an ingress edge node. The main functions in the respective modules are explained below.

Figure 3.7 State-chart diagram of the performance measurement meter in Ingress LER

Upon a packet arrival the state changes from Idle to Arrival. In the Arrival state, the meter classifies and counts the packets and applies the TSW algorithm to calculate the average rate. When the predefined time interval is reached, the Sample Size Estimation state that handles Sample Size Estimation algorithm is activated. Based on the estimated sample size during the time interval, each group (time interval) can be divided into several subgroups. The estimated sample size is equally allocated at each subgroup. Finally, a random number is generated to activate Probe Send state, which inserts the timestamp, a sequence number, etc information into a monitoring packet and sends the monitoring packet into the user traffic stream. The flowchart diagrams of Arrival state, Sample Size Estimation state are depicted in Figure 3.8 and 3.9 respectively. The flow diagram in Figure 3.10 illustrates the monitoring functions in an egress edge router. A router, acting as exit-monitoring node, removes all labels and detects the monitoring packets based on the fields in the packet header. The performance metrics (delay, delay jitter and loss ratio) are computed according to information carried by the monitoring packets. Data packets are

then forwarded using just the traditional IP forwarding paradigm to their respective destinations and the monitoring packets are destroyed.

Figure 3.8 Arrival state flow chart diagram

Figure 3.9 Sample Size Estimation state flow chart diagram

Figure 3.10 Flow diagram of the performance measurement meter in Egress LER

3.5 Simulation Model Implementation

3.5.1 Model Translation

This step is the translation of the operational model into a computer-recognizable form -- the computerized model. In the present work, we implemented the simulation models, using the Optimized Network Engineering Tool (OPNET) [48], since OPNET has many advanced features for network modeling and simulation. The simulator allows us to test the functionality of the adaptive sampling measuring MPLS-based IP network performance under a variety of traffic conditions. Our simulator implements the main components of the system model: the traffic modules, MPLS network module, and the performance measurement meters. These components are mapped into the OPNET constructs: the Network model, the Node model, and the Process model. All are described as follows.

3.5.1.1 *Network Model*

The network model can represent any arbitrary network topology, such as the Ring network, Mesh-torus network and MPLS-based IP network. Figure 3.11 shows the competing traffic network model as described in Section 3.3.2. The network domain is the highest modeling level in OPNET in the sense that it encompasses objects that are defined in the other modeling domains. The network model therefore specifies the entire system to be simulated. A network model contains only three fundamental types of objects: subnetworks, nodes, and links. There are several varieties of nodes and links offering different basic capabilities. The nodes are connected with each other through links. Each node and link has their own identity in the network and can be distinguished from other nodes and links. Each node has the indexed ports to connect with different links.

Figure 3.11 Competing traffic network model

3.5.1.2 Node Model

The node model is used to specify the structure of device models. These device models can be translated as the node objects in the network domain (e.g., computers, packet switch, gateway, etc) As showed in Figure 3.12, a LSR can be translated as a node object in the network model illustrated in Figure 3.11.

The node model is used for directing the connection through different ports of the same node. As showed in Figure3.12, simplex point-to-point links define a connection from a transmitter in the source node to a receiver in the destination node. The packets are conveyed in that one direction. As one of receivers receives a packet, the receiver will forward this packet to the processor. After the processing of the received packet, the processor will either forward this packet or generate new packets to the transmitter.

Figure 3.12 Label switching router node model

3.5.1.3 Process Model

Process models use a Finite State Machine (FSM) paradigm to express the behavior that depends on the current state and new stimuli. The states of the process and the transitions between them are depicted as graphical objects. Process models are instantiated as processes in the node domain and exist within processor or queue modules. Processes can independently execute threads of control that perform general communications and data processing functions. Figure 3.13 illustrates the monitoring procedure in an ingress edge node. The process model comprises 6 states: the Initial state (init), the idle state (idle), the arrival state (arrival), the sample size estimation (sample_est), the probe send state (probe) and the end state (end). The processes of each state are matched to the functions described in Section 3.4.

Figure 3.13 Adaptive sampling process state model

3.5.2 Implementation Validation

Validation is concerned with building the right model. Validation is the determination that a model is an accurate representation of the real system. Validation is usually achieved through the calibration of the model, an iterative process of comparing the model to actual system behavior and using the discrepancies between the two, and insight gained, to improve the model. This process repeated until model accuracy is judged acceptable. As an aid in the validation process, three approaches have been widely followed:

- **Basic function**

The function of the block size in the adaptive sampling process is to change the sampling rate, which is the parameter to determine the number of monitoring packets during the measurement period. So the number of the injected monitoring packets is more

with the decreased block size or less with the increased block size. And the simulation output data just show the expected performance.

- **Animation**

Through the animation feature of OPNET, the operation of the model can be seen visually. The flow of voice and data application traffic can be displayed in the screen. There is no mistake shown in the animation.

- **Analysis of output data graph**

After the simulation, some statistics are recorded in the file and they can be processed and shown in the graphs, such as the status and the number of the monitoring packets and the delay and delay variation estimated by the monitoring packets. By analyzing these graphs, mistakes can be found by examining the changing of the traffic load or the buffers in the simulation time. For example, if the traffic load is heavy, the loss ratio computed and delay estimated by the monitoring packets should increase. If the buffer size is reduced, the loss ratio should increase and the delay should decrease.

Chapter 4

ANALYSIS OF SIMULATION

RESULTS

4.1 Introduction

Network performance is studied using three metrics: (i) end-to-end delay, (ii) delay jitter, (iii) loss ratio. Simulation results are presented for system scenarios differentiated by the following factors:

- Traffic type and traffic characteristics
- Channel bandwidth
- Buffer sizes at label switch routers

Cochran [18] and Krishnaiah and Rao [19] provide some comparative analyses of which sampling strategies offer lower variance under given conditions. These analyses use the variance of the estimate of the mean as a metric for the sampling method; the lower the expected variance of the estimate, the more efficient the sampling method. In our cases we are more interested in assessment of network performance metrics estimated by the sampling methods. Nonetheless we offer some preliminary insights based on this

evaluation mechanism. We use the variance of the estimates to quantify the estimation accuracy. The variance of the estimates can be used to calculate a confidence interval and describe how the estimated means for multiple runs would vary. A higher variance corresponds to a larger confidence interval and a lower accuracy.

In general, results with more samples have lower errors. However, as mentioned earlier, the injected test traffic will affect user traffic characteristics. So the fewer the number of monitoring packets inserted, the lower the effect on the network. The goal of this study is to investigate how well adaptive sampling can reduce the number of sampled packets that are required to achieve a desired level of accuracy, or conversely, increase the accuracy while maintaining the same sample size. We will show that the improvement in precision is achieved by adaptively adjusting the sampling rate based on actual traffic rate. This makes it possible to meet a given accuracy requirement by sampling much fewer packets than other sampling techniques would require.

Simulations are carried out to evaluate the performance of different sampling algorithms under two MPLS-IP based network models. The proposed adaptive sampling and two conventional sampling methods (systematic and stratified sampling) have been evaluated. The comparisons of different sampling techniques are conducted with equal block size so that the results are comparable. A detailed description of these schemes can be found in Chapter 3. We use three popular traffic models: 1) voice sources, 2) video sources, and 3) Internet data sources.

4.2 Performance Evaluation of the Adaptive, Systematic and Stratified Sampling Methods

4.2.1 Simple Topology for Voice

We design the simple network topology to verify the operations of the adaptive sampling method and evaluate the performance of this method by comparing it to the conventional sampling methods. The adaptive, systematic and stratified sampling methods are then simulated and evaluated by the resolution of the measurements (the block size described in Chapter 2 is set to 200, 300, 400, 500, 600). In the experiments, we use voice traffic as the network input source. The voice traffic model and simple network model are described in Chapter 3. The voice-sampling rate is set to 64 Kbps, and an active voice user generates one packet per 16ms. Each packet contains 172 speech bytes. There are 50 voice users generating about 1.5 Mbps aggregated traffic load from source to destination. In this simulation, the length of simulation runs is defined as 10 minutes, the time interval is defined as 2 seconds, and the number of replication is defined as 20. The simulation independent replications are made. For each experiment the mean value and the standard deviation for the estimation results from the 20 rounds are calculated and compared with the actual value.

Delays are determined by the waiting times in the FIFO queue and the service times. They are characterized by their mean values. In Figure 4.1 values are plotted against the fixed block size, along with their respective 95% confidence interval. We observe in this experiment that the true value of the estimated quantity lies within or very close to the confidence interval for the adaptive sampling without exception. The confidence interval is

increased as the block size increases. Nevertheless, the adaptive sampling clearly gives more accurate estimated results than conventional sampling, but this better accuracy starts deteriorating when the block size becomes large.

Figure 4.1 Delay vs. Block size – voice traffic

In Figure 4.2 delay jitters are plotted against different block sizes, along with their respective 95% confidence interval. With small block sizes, the values of the adaptive sampling and the stratified sampling methods provide a good fit for the estimated points. The confidence interval for the adaptive sampling is smaller than that for the stratified sampling in all cases, but this match starts getting worse when the block size exceeds 600 packets.

Figure 4.2 Delay jitter vs. Block size – voice traffic

Figure 4.3 shows the loss ratios computed by the monitoring packets with these three different sampling methods. Compared with the true value (which is 0.8% in this test), the loss ratio of the systematic sampling is quite a bit higher, while adaptive and stratified sampling achieves values almost equal to the true value. Thus, the loss ratio can be estimated quite closely by simply counting the adaptive and stratified samples instead of all the packets.

The above simulation results clearly confirm our design goal for adaptive sampling, as discussed in Chapter 2. Thus, adaptive sampling can provide more precise estimations on delay and delay variation compared with stratified and systematic sampling using the same block size under voice traffic.

Figure 4.3 Loss ratio vs. Block size – voice traffic

4.2.2 Simple Topology for Video Traffic

We replace voice traffic with video traffic and set the traffic load as 2.5 Mbps. The video source traffic model is described in Chapter 3. In this simulation, the length of simulation runs is defined as 10 minutes, the time interval is defined as 1 second, and the number of replication is defined as 20. The simulation independent replications are made. The adaptive, systematic and stratified sampling methods are then simulated and evaluated by the resolution of the measurements (the block size described in Chapter 2 is set to 200, 300, 400, 500, and 600).

As before, Figures 4.4 and 4.5 display the measurements of estimation accuracy for delay and delay jitter. Figure 4.6 displays the loss ratio computed by the monitoring packets using these three different sampling methods. We observe in this experiment that the true value of the estimated quantity lies within the confidence interval for all three sampling methods without exception. The results in these experiments with video traffic are different from those with voice traffic. The adaptive sampling method achieves a slightly lower loss rate and slightly smaller error rates.

When dealing with traffic of a periodic nature, as is the case with video traffic, the systematic sampling method is effective, provided that the appropriate sampling interval can be identified and employed. Therefore, the primary goal of using adaptive sampling for periodic traffic is to match the performance of the best systematic sampling rate. An advantage of adaptive sampling is that this optimal rate can be adjusted dynamically if the frequency or type of periodic traffic changes.

Figure 4.4 Delay vs. Block size – video traffic

Figure 4.5 Delay jitter vs. Block size – video traffic

Figure 4.6 Loss ratio vs. Block size – video traffic

4.2.3 Simple Topology for Self-similar Traffic

The self-similar traffic model described in Chapter 3 is utilized in this test. One feature of self-similar traffic is its long-range dependence, which implies the presence of burstiness at all time scales. In the simulation, by setting alpha-stable $\alpha = 1.95$ and $H = 0.834$, the bursty traffic is obtained. The users' traffic model generates 2.7Mbps traffic load. The length of simulation runs is defined as 10 minutes, the time interval is defined as 1 second, and the number of replication is defined as 20. The simulation independent replications are made. The adaptive, systematic and stratified sampling methods are then simulated and evaluated by the resolution of the measurements (the block size described in Chapter 2 is set to 200, 300, 400, 500, and 600).

Figure 4.7 Delay vs. Block size – self-similar traffic

Figure 4.8 Delay jitter vs. Block size – self-similar traffic

In Figures 4.7 and 4.8, the estimated values are plotted against the fixed block size, along with their respective 95% confidence interval. We observe in this experiment that the true value of the estimated quantity lies within or very close to the confidence interval for adaptive and stratified sampling without exception. The confidence interval for adaptive sampling is smaller than that for stratified sampling in all cases, but this match starts getting worse when the block size exceeds 500 packets.

Figure 4.9 shows the loss ratio computed by the monitoring packets with these three different sampling methods. Compared with the true value (which is 0.5% in this test), the loss ratio of the systematic sampling method is substantially higher, while the adaptive and the stratified sampling methods maintain an almost fixed value approximate to the theory. Thus, the loss ratio can be estimated by simply counting the adaptive samples instead of all the packets.

Figure 4.9 Loss ratio vs. Block size – self-similar traffic

4.2.4 Competing Traffic for Voice

The competing traffic model is developed to demonstrate the effects of competing traffic on the performance of the sampling methods. In the experiments, we use three aggregated voice traffic as the network input source, each one of them is aggregated from 50 users, and sent from Edge LSR1 to the destination nodes LSR2, LSR3, and LSR4 respectively along their LSPs. The traffic from LSR1 to LSR3 is called the monitored traffic, since the monitoring packets are inserted based on this traffic. The competing traffic is sent from LSR4 to LSR3, passing through LSR2. The monitored traffic and the competing traffic are merged at LSR2. With the volume of competing traffic increasing, the network performance under the adaptive and stratified sampling methods are simulated and evaluated with different block sizes. In this simulation, the simulation time is set as 10

minutes, the time interval is defined as 2 seconds, and the number of replication is defined as 20. Assume in the simulation independent replications are made.

Figures 4.10, 4.11, 4.12 and 4.13 show the delay and the delay jitter with a 95% confidence interval estimated by the adaptive and the stratified sampling while the block size set as 300 and 500 respectively. As the competing traffic increases, the mean values of the delay are increased. We observe that in this experiment, even when the block size is 500, the true value of the estimated quantity lies within the confidence interval for the adaptive and stratified sampling without exception. The confidence interval for adaptive sampling is smaller than that for stratified sampling. Thus, adaptive sampling can provide more precise estimations on delay and delay variation under the effect of the competing traffic.

Figure 4.10 Delay vs. Competing traffic load (block size = 300)

Figure 4.11 Delay jitter vs. Competing traffic load (block size = 300)

Figure 4.12 Delay vs. Competing traffic load (block size = 500)

Figure 4.13 Delay jitter vs. Competing traffic load (block size = 500)

Figure 4.14 and 4.15 show the loss ratio computed by the monitoring packets with these two different sampling methods. As the competing traffic increases, the loss ratio of the theory is changed from 0.8% to 5.8%. Compared with the true values, the loss ratio of the stratified and the adaptive sampling methods maintain a value almost the same as the true value. Thus, the loss ratio can still be estimated by simply counting the adaptive and stratified samples instead of all the packets under the effect of the competing traffic.

Figure 4.14 Loss ratio vs. Competing traffic load (block size = 300)

Figure 4.15 Loss ratio vs. Competing traffic load (block size = 500)

4.2.5 Competing Traffic for Self-similar Traffic

We replace voice traffic with self-similar traffic in this experiment. Each aggregated traffic load is set as 2.7 Mbps. With the increase in volume in competing traffic, the network performance under the adaptive and stratified sampling methods were simulated and evaluated with different block sizes. In this simulation, the length of simulation runs is defined as 10 minutes, a number of batched is used by 600 with equal time interval 1 second, and the number of replication is defined as 20. Assume in the simulation independent replications are made.

In Figures 4.16, 4.17, 4.18 and 4.19, the estimated delay and delay jitter values are plotted against the competing traffic volumes, along with their respective 95% confidence intervals while the block size set as 300 and 500 respectively. As the competing traffic increases, the mean values of the delay are increased. We observe that in this experiment, even the block size is 500, the true value of the estimated quantity lies within the confidence interval for adaptive and stratified sampling without exception. The confidence interval for adaptive sampling is smaller than that for stratified sampling. Thus, the adaptive sampling can provide more precise estimations on delay and delay variation under the effect of the competing traffic.

Figure 4.20 and 4.21 show the loss ratio computed by the monitoring packets with these two different sampling methods. As the competing traffic increases, the loss ratio of the theory is changed from 0.5% to 2.8%. Compared with the true values, the loss ratio of the stratified and the adaptive sampling maintains a value almost the same as the true value.

Thus, the loss ratio can still be estimated by simply counting the adaptive and stratified samples instead of all the packets under the effect of the competing traffic.

Figure 4.16 Delay vs. Competing traffic load (block size = 300)

Figure 4.17 Delay jitter vs. Competing traffic load (block size = 300)

Figure 4.18 Delay vs. Competing traffic load (block size = 500)

Figure 4.19 Delay jitter vs. Competing traffic load (block size = 500)

Figure 4.20 Loss ratio vs. Competing traffic load (block size = 300)

Figure 4.21 Loss ratio vs. Competing traffic load (block size = 500)

4.3 Factors Impacting on the Network Performance

We studied the performance of different sampling methods by applying different traffic types and characteristics, as discussed above. The buffer sizes and utilizations are the key parameters for quantifying the effect on the network performance monitored by the adaptive sampling techniques. The simulation results presented analyze the effect of each of the following factors.

4.3.1 Effect of Traffic Utilization

In order to acquire a clearer idea of how different traffic utilization affects the network performance monitored by the adaptive sampling method, we are presenting the mean value and the standard deviation to assess the accuracy of the sampling / estimation procedure for delay under different traffic utilizations. The voice, self-similar traffic model

and simple network model described in Chapter 3 are utilized in this test. In the first set of experiments, we use voice traffic as the network input source and set voice traffic utilization as 40%, 45%, 50%, 55% and 60% respectively. We define the length of simulation runs as 10 minutes, and the time interval as 2 seconds. In the second set of experiments, we replace voice traffic with self-similar traffic and keep the same parameters. Figure 4.22, 4.23, 4.24, and 4.25 show the estimated values of the mean delay and the standard deviation against the traffic utilization under the different block sizes.

Figure 4.22 Delay mean vs. Utilization – voice traffic

Figure 4.23 Delay STD vs. Utilization – voice traffic

Figure 4.24 Delay mean vs. Utilization – self-similar traffic

Figure 4.25 Delay STD vs. Utilization – self-similar traffic

By comparing the results under different traffic utilization, we can find that in any input traffic sources, for simple network model, the higher traffic utilization, the more significant effect on the network performance. As a result, the adaptive sampling method performed well, even in the worst case (block size = 600), the estimated values lies exactly on the reference line (actual value). In fact, the estimated values are not affected by a reduction of the data from which it is computed. Hence, sampling gives a definitely accurate way for delay monitoring. Network operators will choose the sampling rate depending on the required timeliness of the results. The higher sampling rate the more timely results available, but more resources are necessary.

4.3.2 Effect of Buffer Size

The end-to-end delay in the network is dominated by queuing delay. Large buffer sizes can cause large delay and delay jitter. In order to balance losses and delay, keeping the buffer size at a reasonable level is important. We investigate the buffer size on the network performance monitored by the adaptive sampling method. Our analysis addresses the mean value and the standard deviation of delay to assess the accuracy of the sampling procedure. In order to get a better idea how the buffer size at a core router affects the network performance, different buffer sizes, different traffic characteristics and different block sizes are considered and used in the experiments. The traffic used in this set of experiments is voice traffic and self-similar traffic. In the first set of experiments, the effect of different buffer sizes on the network performance under voice traffic is examined. The voice traffic load is set as 1.5 Mbps. The buffer size is set as 50, 75, 100, 125 and 150 respectively. In the second set of experiments, we replace voice traffic with self-similar traffic and the traffic load is set as 2.7 Mbps. The buffer size is set as 50, 100, 150, 200, and 250 respectively. The figures generated through the simulations are presented as below.

Figure 4.26 Delay mean vs. Buffer size – voice traffic

Figure 4.27 Delay STD vs. Buffer size – voice traffic

Figure 4.28 Delay mean vs. Buffer size – self-similar traffic

Figure 4.29 Delay STD vs. Buffer size – self-similar traffic

Figure 4.26, 4.27, 4.28 and 4.29 show the effect of different buffer sizes on the performance of the network parameter (delay) estimated by adaptive sampling with different block sizes. As expected, the mean value and the standard deviation of delay increase considerably by increasing the buffer size. We find that using two traffic sources, the effect of the buffer size on the network performance is similar. The higher buffer size, the higher delay. The adaptive sampling method with different block sizes can provide more accurate estimated values, but the accuracy of this estimation is impaired when the block size increases.

4.4 Summary of Simulation Results

In this chapter, we have evaluated the performance of adaptive sampling to achieve gains in precision or efficiency. Compared with the conventional sampling methods, the advantages of the adaptive sampling are presented through a set of numerical results. In brief, adaptive sampling performs better than all other methods on voice traffic, as well as other methods on video traffic, and slightly worse than the stratified method on bursty self-similar traffic. Furthermore, we demonstrated that with the competing traffic model, the competing traffic has negligible less impact on the sampling results when evaluating network performance. Finally, by investigating some factors, which many impact the network performance, the adaptive sampling method can give a definitely accurate way for delay monitoring. It demonstrates, once more, that the adaptive sampling method provides the best potential for monitoring and managing high-performance networks due to its adaptability to the traffic rate and traffic type.

Chapter 5

CONCLUSIONS

5.1 Conclusions

The explosion of the Internet will demand proper monitoring and measurement methods in order to make network design, capacity planning and forecasting, operations and management effective and timely. This dissertation has presented a new adaptive sampling scheme for monitoring and measuring network performance metrics in MPLS-based IP networks supporting voice, video and real-time data applications. We are focusing on developing the adaptive sampling techniques into monitoring and control functions aims to make them less invasive and more efficient over the future Internet.

We address issues related to how the adaptive sampling technique achieves the accuracy of the network performance while maintaining the same sample counts or reduction of samples by comparing it to the conventional sampling methods. We propose the improvements of this method under different behaviors of the applications. A variety of parameters were monitored, such as packet losses, packet one-way delay and delay variation at the routers. In terms of two MPLS-based IP network topologies, one is the simple network topology used to verify the operations of the adaptive sampling method.

Another one is the competing traffic topology used to demonstrate the effects of the competing traffic on the performance of the adaptive sampling method.

Our work produced some interesting results. The adaptive technique was shown to perform well on voice traffic and Internet data traffic. This approach is able to achieve better accuracy in estimating the average delay, delay variation and packet losses, while maintaining the same sample counts compared with the conventional sampling techniques based on the same block size. The results of delay depend on the number of samples (i.e. the monitoring packets). Accurate measurements are required to find bottlenecks, while at the same time the impact on the network must be minimized to allow the applications to take advantage of the low latency and loss ratio such networks provide. For periodic data such as a video traffic pattern, the adaptive sampling approaches perform comparably to the best systematic approach in terms of accuracy and the sample count. However, perhaps more importantly, the adaptive techniques have the ability to adjust to changing traffic loads, and therefore in an environment with dynamic traffic patterns, such as with data communications, multimedia and integrated services, the adaptive sampling techniques hold the potential to outperform systematic sampling with its pre-selected and fixed sampling interval.

Moreover, this approach has a number of desirable properties. Firstly, the simple processing of the monitoring packets gives direct measurements of end-to-end path characteristics, such as delay and packet loss rate. Its advantage is that it does not require any measurement support from the network. Secondly, with the competing traffic increasing, the method provides acceptable estimates of the network performance

parameters. The competing traffic has limited effect on the accuracy of this approach to measure end-to-end path characteristics for MPLS-based IP networks.

5.2 Future Work

While a considerable amount of effort was made to acquire the results presented in this thesis, there is no doubt many questions were not answered. We provide below a few suggestions of future research, coming as continuation of our work.

The implemented adaptive sampling method is quite simple and has therefore a number of limitations and shortcomings that can be improved. We can use more precise traffic prediction methods such as fuzzy logical or linear prediction algorithm instead of the Time Sliding Window algorithm. In the future, different stratification strategies can be investigated with different allocation schemes, different number of strata and different methods for the calculation of the boundaries to the varying trace characteristics. Furthermore, better methods need to be investigated to predict those characteristics for the subsequent segment.

In our work, we addressed the traffic models to simulate the real-time traffic patterns and evaluated the performance of the adaptive sampling method using the simulation tool. However, in the future, experiments should be performed with real traffic traces, and the prototype is emulated and the main application is monitoring in real networks.

Moreover, we can investigate other potential adaptive schemes for self-similar traffic. Also the system that we developed can be expanded by incorporating MPLS capabilities in its structure using signaling protocols like RSVP, LDP or CR-LDP rather than using the

static LSPs. By doing so, a new set of architectures can be established and tested, generating a new avenue for interesting research.

REFERENCES

- [1] IETF, <http://www.ietf.org>
- [2] S. Floyd, V. Paxson, "An Overview of Internet Engineering, Measurements and Modeling", ITC15 Tutorial, Washington DC, 22 June 1997
- [3] Bhattacharyya, "Network Measurement and Monitoring: A Sprint Perspective", Internet Draft, <draft-bhattacharyya-monitoring-sprint-00.txt>
- [4] Advanced Network & Services, Inc., <http://www.Advanced.org>
- [5] CAIDA Measurement Tool Taxonomy, <http://www.caida.org>
- [6] Internet2 Working Groups, www.internet2.edu/html/working_groups.html
- [7] National Laboratory for Applied Network Research (NLNR), <http://moat.nlanr.net>
- [8] V. Paxson, G. Almes, J. Mahdavi, and M. Mathis, "Framework for IP Performance Metrics," Request for Comments 2330, May 1998
- [9] W. S. Lai. "A Framework for Internet Traffic Engineering Measurement", Internet Draft, <draft-ietf-tewg-measure-02.txt>
- [10] ITU-T Recommendation I.350 "General Aspects of Quality of Service and Network Performance in Digital Networks" includes ISDN. March 1993
- [11] ITU-T Recommendation I.380 "Internet Protocol Data Communication Service-IP Packet Transfer and Availability Performance Parameters", February 1999
- [12] G. Almes. S. Kalidindi, M. Zekauskas, "A One-way Delay Metric for IPPM", RFC-2679, IETF, September 1999
- [13] C. Demichelis, P. Chimento, "IP Packet Delay Variation Metric for IPPM", Internet Draft, <Draft-ietf-ippm-ipdv-07.txt>, February 2001
- [14] G. Almes. S. Kalidindi, M. Zekauskas, "A One-way Packet Loss Metric for IPPM", RFC-2680, IETF, September 1999
- [15] D. J. Goodman., "Efficiency of Packet Reservation Multiple Access", IEEE Trans. on Veh. Tech., Vol. 40, No. 1, Feb. 1991, pp. 170-176
- [16] G. Almes. S. Kalidindi, M. Zekauskas, "One-way Loss Pattern Sample Metrics", RFC-2681, IETF, September 2000

- [17] T. Lindh, "A New Approach to Performance Monitoring in IP Networks – combining active and passive methods", Proceedings of Passive and Active Measurement Workshop (PAM 2002)
- [18] W. G. Cochran, "Sampling Techniques", Wiley, New York, 1977
- [19] P.R. Krishnaiah, C.R. Rao, "Handbook of Statistics", Vol. 6: Sampling, North-Holland, Amsterdam, 1988
- [20] Packet Sampling Working Group <https://ops.ietf.org/lists/psamp>
- [21] K.C. Claffy, G.C. Polyzos, H. Braun, "Application of Sampling Methodologies to Network Traffic Characterization", Proceedings of ACM SIGCOMM'93, San Francisco, CA, USA, September 13 - 17, 1993
- [22] NetFlow Services and Applications, White Paper, Cisco Systems, 1999
- [23] N. Duffield, M. Grossglauser, "Trajectory Sampling for Direct Traffic Observation", Proceedings of ACM SIGCOMM 2000, Stockholm, Sweden, August 28 - September 1, 2000
- [24] I. Cozzani, S. Giordano, "Traffic Sampling Methods for End-to-end QoS Evaluation in Large Heterogeneous Networks", Computer Networks and ISDN Systems, 30(16-18), September 1998
- [25] T. Zseby, "Deployment of Sampling Methods for SLA Validation with Non-Intrusive Measurements ", Proceedings of Passive and Active Measurement Workshop (PAM 2002)
- [26] T. Zseby, S. Zander, G. Carle, " Evaluation of Building Blocks for Passive One-way-delay Measurements", Proceedings of Passive and Active Measurement Workshop (PAM 2001)
- [27] T. Lindh, "An Architecture for Embedded Monitoring of QoS Parameters in IP Based Virtual Private Networks", Proceedings of Passive and Active Measurement Workshop (PAM 2001)
- [28] B. Choi, J. Park, Z.L. Zhang, "Adaptive Random Sampling for Load Change Detection", Extended Abstract in Proc. ACM SIGMETRICS June 2002
- [29] W.H. Ma, J. Yan, C.C. Huang, "Adaptive Sampling Methods for Network Performance Metrics Measurement and Evaluation in MPLS-based IP Networks", CCECE'03 May 2003

- [30] S. Floyd, V. Jacobson, "The Synchronization of Periodic Routing Messages",
IEEE/ACM Transactions on Networking, 2(2), pp.122-136, April 1994
- [31] T. Zseby, "Stratification Strategies for Sampling-based Non-intrusive
Measurements of One-way Delay", Proceedings of Passive and Active
Measurement Workshop (PAM 2003)
- [32] P.D. Amer, L.N.Cassel, "Management of Sampled Real-Time Network
Measurements", 14th Conference on Local Computer Networks, October 1989,
Minneapolis, pages 62-68, IEEE 1989
- [33] D. D. Clark, W. J. Fang, "Explicit Allocation of Best-Effort Packet Delivery
Service", IEEE/ACM Transactions on networking, August 1998
- [34] W. Fang, N. Seddigh, B. Nandy, "Time Sliding Window Three Color Marker,"
RFC2859, IETF, March 2000
- [35] J. Banks, J. S. Carson, B. L. Nelson, "Discrete-Event System Simulation", (2nd)
Prentice Hall, Upper Saddle River, NJ, 1996
- [36] V. Paxson, S. Floyd, "Wide Area Traffic: The Failure of Poisson Modeling",
IEEE/ACM Trans. On Networking, Vol.3, NO.3. June 1995
- [37] E. Rosen, A. Viswanathan, R. Callon, "Multiprotocol Label Switching
Architecture", RFC-3031, January 2001
- [38] R. Callon, P. Doolan, N. Feldman, A. Fredette, G. Swallow, A. Viswanathan,
"A Framework for Multiprotocol Label Switching", September 1999
- [39] B. Davie and Y. Rekhter, "MPLS: Technology and Applications", Morgan
Kaufmann, 2000
- [40] E. N. Gilbert, "Capacity of a Burst-noise Channel", Bell Syst. Tech. J., pp. 1253,
September 1960
- [41] D. Goodman, S. Wei, "Efficiency of Packet Reservation Multiple Access",
IEEE Transactions on Vehicular Technology, Vol. 40, No. 1, pp 170-176,
February 1991
- [42] Recommendation H.261, ITU-T Standard, March 1993
- [43] Recommendation H.263, ITU-T Standard, February 2000
- [44] B. Tsybakov, N. D Georganas, "On Self-Similar Traffic in ATM Queues:
Definitions, Overflow Probability Bound, and Cell Delay Distribution",

IEEE/ACM Transactions on Networking, Vol. 5, No. 3 (June), pp 397-408,
1997

- [45] W. Willinger, M. S. Taqqu, R. Sherman, D. V Wilson, "Self-Similarity Through High-Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level", IEEE/ACM Transactions on Networking, Vol. 5, No. 1 (Feb.), pp 71-86, 1997
- [46] I. Norros, "On the Use of Fractional Brownian Motion in the Theory of Connectionless Networks", IEEE JSAC, Vol. 13, No. 6 (Aug.), pp 953-962, 1995
- [47] J.R. Gallardo, D. Makrakis, L. Orozco-Barbosa, " Use of Alpha-stable Self-similar Stochastic Process for Modeling Traffic in Broadband Networks", Performance Evaluation, 40(1-3), pp. 71-98, 2000
- [48] <http://www.opnet.com>