A Quality of Service Monitoring System for Service Level Agreement Verification

Xiaoyuan Ta

A thesis submitted in fulfilment of the
requirements for the award of the degree of

MASTER OF ENGINEERING BY RESEARCH

SCHOOL OF ELECTRICAL AND INFORMATION ENGINEERING
THE UNIVERSITY OF SYDNEY

March 2006

Abstract

Service-level-agreement (SLA) monitoring measures network Quality-of-Service (QoS) parameters to evaluate whether the service performance complies with the SLAs. It is becoming increasingly important for both Internet service providers (ISPs) and their customers. However, the rapid expansion of the Internet makes SLA monitoring a challenging task. As an efficient method to reduce both complexity and overheads for QoS measurements, sampling techniques have been used in SLA monitoring systems.

In this thesis, I conduct a comprehensive study of sampling methods for network QoS measurements. I develop an efficient sampling strategy, which makes the measurements less intrusive and more efficient, and I design a network performance monitoring software, which monitors such QoS parameters as packet delay, packet loss and jitter for SLA monitoring and verification.

The thesis starts with a discussion on the characteristics of QoS metrics related to the design of the monitoring system and the challenges in monitoring these metrics. Major measurement methodologies for monitoring these metrics are introduced. Existing monitoring systems can be broadly classified into two categories: active and passive measurements. The advantages and disadvantages of both methodologies are discussed and an active measurement methodology is chosen to realise the monitoring system.

Secondly, the thesis describes the most common sampling techniques, such as systematic sampling, Poisson sampling and stratified random sampling. Theoretical analysis is performed on the fundamental limits of sampling accuracy. Theoretical

analysis is also conducted on the performance of the sampling techniques, which is validated using simulation with real traffic. Both theoretical analysis and simulation results show that the stratified random sampling with optimum allocation achieves the best performance, compared with the other sampling methods. However, stratified sampling with optimum allocation requires extra statistics from the parent traffic traces, which cannot be obtained in real applications. In order to overcome this shortcoming, a novel adaptive stratified sampling strategy is proposed, based on stratified sampling with optimum allocation. A least-mean-square (LMS) linear prediction algorithm is employed to predict the required statistics from the past observations. Simulation results show that the proposed adaptive stratified sampling method closely approaches the performance of the stratified sampling with optimum allocation.

Finally, a detailed introduction to the SLA monitoring software design is presented. Measurement results are displayed which calibrate systematic error in the measurements. Measurements between various remote sites have demonstrated impressively good QoS provided by Australian ISPs for premium services.

Statement of Originality

I hereby declare that this thesis, submitted in fulfilment of the requirements for the award of Master of Engineering by Research, in the school of Electrical and Information Engineering, the University of Sydney, is my own work unless otherwise referenced or acknowledged. The document has not been accepted for the award of any other qualification at any educational institution.

Signed

Xiaoyuan Ta
March 2006

Acknowledgments

First and foremost, I would like to express my appreciation to my supervisor, Dr Guoqiang Mao, for his kind advice and support during my study. He has not only supplied an interesting research topic but also made many contributions and valuable comments. Moreover, he patiently read and marked up my draft thesis and gave me valuable suggestions that significantly improved the quality of this thesis. I have learned a lot from him, especially on how to be a good researcher—being rigorous, energetic and dedicated to research.

I would also like to thank Mr Lixiang Xiong and Mr Zhuo Chen, for their kind help and suggestions for revising this thesis. I would also like to thank people in the telecommunication research group, for their constant and ongoing generous help.

I would like to express my gratitude to my family. My parents and my sister always give me support, care and love. My wife's parents always give me understanding and encouragement. Their passion for careers and their attitudes to life are always role models for me. I owe a special gratitude to my wife Ying Shan. Her constant love, support and inspiration have made my Master's study such an enjoyable and rewarding experience.

The work presented in this thesis has been supported by the contracted research project "BLO No. 7260" from Optus. I am grateful to Optus for providing financial support for my study.

Contents

List of Figures

List of Tables

Acronyms

ACF	Autocorrelation Function
CDF	Cumulative Distribution Function
CPU	Central Processing Unit
DF	Distribution Function
fpc	Finite Population Correlation
GUI	Graphical User Interface
ISP	Internet Service Provider
ITU	International Telecommunications Union
IETF	Internet Engineering Task Force
IPPM	IP Performance Metrics Working Group
ICMP	Internet Control Message Protocol
LMS	Least Mean Square
LP	Linear Prediction
LAN	Local Area Network
MSE	Mean Square Error
MBZ	Must Be Zero
NTP	Network Time Protocol
NLR	Noticeable Loss Rate
NIC	Network Interface Card
PDF	Probability Density Function
QoS	Quality of Service
RFC	Requests For Comments
RTT	Round-trip Time
RTO	Retransmission Timeout

SLA	Service Level Agreement
SDU	Service Data Unit
SNMP	Simple Network Management Protocol
TCP	Transmission Control Protocol
TOS	Type of Service
UDP	User Datagram Protocol
UI	User Interface
VoIP	Voice over Internet Protocol
WAN	Wide Area Network

Chapter 1

Introduction

1.1 Background

Internet Service Providers (ISPs) now offer service level agreements (SLAs) routinely to their customers. Management needs contractual guarantees that business objectives are met, and end-users demand assurance that their critical network applications and services are available when needed. The availability of SLAs and a means to validate them gives management the confidence to move ahead. The wide adoption of the E-business model has made it essential that service-providers deliver on SLAs in a quantitative and qualitative manner. This has driven the service-providers to seek consistent testing and measurement methods that make real sense of customer network performance.

An SLA is defined by the International Telecommunications Union (ITU) as "a negotiated agreement between a customer and the service provider on levels of service characteristics and the associated set of metrics. The content of SLAs varies depending on the service offering and includes the attributes required for the negotiated agreement" [1]. The Internet Engineering Task Force (IETF) defines SLAs in a similar way [2]. Figure 1.1 shows the main features of the SLAs.

Generally speaking, a good SLA should include these three key aspects:

- Service level objectives: encompass Quality-of-Service (QoS) parameters or

class of service provided, service availability and reliability, authentication issues, SLA expiry date, and so on.

- Service measuring components: specify the way of measuring service quality and other parameters used to assess whether the service complies with the SLA.

- Financial compensation components: include billing options, penalties for breaking the contract, and so forth.

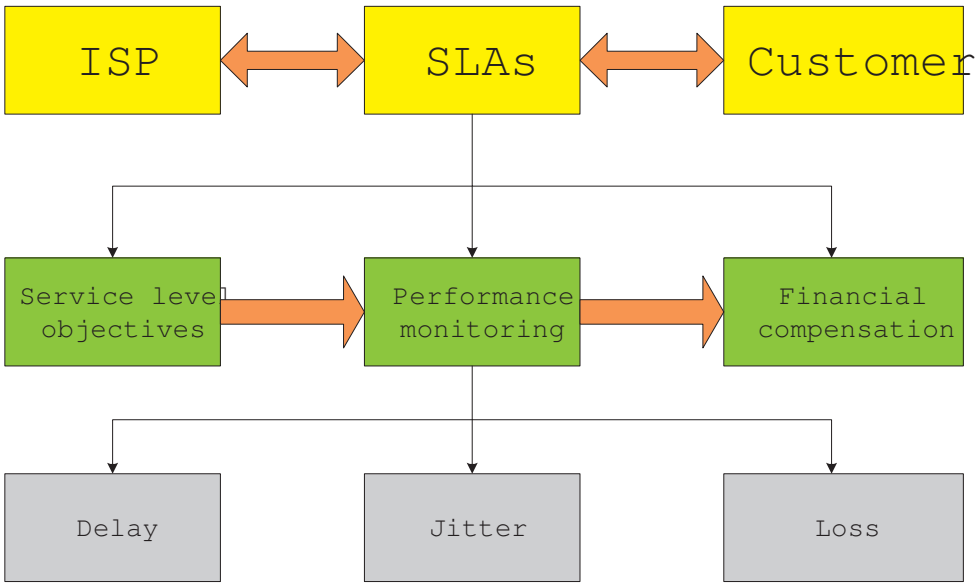


Figure 1.1 Structure of service-level-agreements

SLA monitoring is an important part of SLA management. It is useful for both network operators and individual customers, who want to check whether the service performance indeed complies with the SLAs. Moreover, the ability to measure against key performance indicators facilitates the continuous quality improvement process. It helps the ISPs to locate the bottleneck in their networks. A service performance problem becomes an opportunity to structurally improve overall service quality and customer satisfaction.

1.2 Research Motivation and Contribution

SLA monitoring is about collecting statistical metrics about network performance to evaluate whether the provider complies with the level of QoS that the customer expects [3]. Therefore, accurate measurement and estimation of network performance becomes a key challenge in SLA monitoring. However, the implementation of measurement becomes increasingly difficult and complex due to the rapid expansion of the Internet. Traditional measurement tools, such as “ping”, cannot satisfy the measurement requirements nowadays. Moreover, the dramatic increase in the speed of wide area backbones presents obstacles to complete statistics collection. The enormous amount of measurement data may significantly increase the cost and resource usage [4].

In order to solve these problems, sampling techniques are employed in SLA monitoring systems to reduce the quantity of control data and resources required to process it, and finally to reduce the measurement complexity and cost. Systematic sampling and random sampling are two widely used methods in existing monitoring systems, but both of them have severe limitations. Stratified random sampling can achieve higher estimation accuracy, but its high complexity may compromise its advantages.

The aim of this research project, which has been funded by Optus through the research contract “BLO No. 7260”, is to develop an efficient sampling strategy to make the measurement less intrusive and more efficient. Then a network performance monitoring software, which monitors such QoS parameters as packet delay, packet loss and jitter for SLA monitoring and verification, and which uses the proposed sampling strategy, needs to be designed. These objectives have been fully achieved. Firstly, a theoretical analysis of the performance of different sampling techniques (both count-based and timer-based) is presented. Secondly, a novel adaptive stratified sampling strategy is developed and validated. Finally, QoS monitoring software is delivered at the end of the project, which has been highly rated by Optus. This thesis provides a comprehensive summary of the outcome of the project.

1.3 Thesis Outline

This thesis consists of seven chapters, the rest of which are organised as follows:

Chapter 2 presents a comprehensive review of related work. Firstly, I describe the main usages of Internet measurement, and the standard metrics for measurement as defined by the IETF's IP Performance Metrics Working Group (IPPM). Secondly, I discuss in detail the characteristics of QoS metrics related to the design of the monitoring system in this project, i.e., packet delay, packet loss and jitter, and challenges in monitoring these metrics. Thirdly, I introduce the major methodologies of network performance measurement, including both passive measurement and active measurement, as well as their advantages and disadvantages.

Chapter 3 describes major sampling techniques that can be used in the sampling-based monitoring system, such as systematic sampling, random sampling, stratified random sampling and adaptive sampling. Discussion of the fundamental limit (i.e., minimum sample size required for a given confidence level and an error bound) of the accuracy of sampling techniques is then presented.

Chapter 4 presents a theoretical analysis of the performance of two fundamental sampling techniques, i.e., systematic sampling and random sampling, and compares their performance. Autocorrelation ρ of packet delay of the parent delay trace is used as a factor in the performance comparison between time-based systematic sampling and time-based Poisson sampling. ρ is also used to determine the stratification boundaries for stratified sampling. Simulation results using real traffic trace provided by the WAND group is presented to validate the theoretical analysis.

Chapter 5 proposes an adaptive stratified sampling strategy for SLA monitoring, which is based on the stratified sampling with optimum allocation discussed in Chapter 4.4.2. Although stratified sampling with optimum allocation can achieve a satisfactory accuracy of estimation, it has severe imitations. The stratified sampling with optimum allocation requires extra statistics (e.g., standard deviation of packet delay, total number of packets) of the parent trace to determine the stratum sample size. In real applications, these statistics are not known *a priori*. To address the challenge,

a novel adaptive sampling method is proposed, which employs a least-mean-square (LMS) algorithm to predict the standard deviation of packet delay from past observations. The sample size for the next stratum is calculated from the predicted standard deviation. Sampling results that show good performance are presented.

Chapter 6 provides a detailed introduction to the monitoring software design. I start with an introduction to the software environment and functionality. A description of the procedure of the TCP measurement, UDP measurement and ICMP measurement is then presented. The systematic error of the software is calibrated. Finally, I introduce the software's graphic-user-interface (GUI) design and demonstrate several test results in real networks.

Chapter 7 concludes this thesis by providing a summary of my major contributions. The direction for future study is also discussed.

Chapter 2

Literature Review

Before entering into detailed discussion of sampling techniques and their performance comparison, a review of the relevant work on QoS measurements is presented.

2.1 Characteristics of QoS Metrics

In this section, characteristics of packet delay, packet loss and jitter, which are related to the design of the monitoring system and the challenges in monitoring these metrics are discussed. Firstly, an overview of the main usages of Internet measurements is provided, followed by a brief list of standard metrics, as defined by IETF.

2.1.1 Main Usages of Internet Measurements

As described in [5], the main usages of Internet measurements are Internet topology measurement, workload measurement, performance monitoring and routing measurement.

- Topology measurement: collects information on the network connectivity and graphical locations of network devices. With the rapid development of Internet, it becomes a challenge to track and visualise the complex Internet topology [5].
- Workload measurement: focuses on the collection of information on the resource usage of routers or switches and the link utilisation [5], [6].

- Performance measurement: is used by network users or researchers in analysing traffic behaviour on specific paths or the performance (e.g., packet delay, jitter, packet loss) associated with individual ISPs. A recent development in the industry is the monitoring of SLAs [5].
- Routing measurement: measures the dynamics of routing protocols and routing updates [6].

2.1.2 Standard Metrics

The IETF's IPPM has developed series of standards called Requests For Comments (RFC) on network performance measurements. The standard metrics for measurements are defined in RFC 2330, which are listed below:

- Metric for Measuring Connectivity (RFC2678) [7];
- A One-way Delay Metric (RFC2679) [8];
- A One-way Packet Loss Metric (RFC2680) [9];
- A Round-trip Delay Metric (RFC2681) [10];
- One-way Loss Pattern Sample Metric (RFC 3357) [11];
- IP Packet Delay Variation Metric (RFC 3393) [12].

2.1.3 Packet Delay

Packet delay is the delay experienced by packets when passing through the network. It may be considered either in an end-to-end relation or with regard to a particular network element. SLAs for network delay are generally defined in terms of one-way end-to-end delay for nonadaptive time critical applications (such as VoIP and video) or in terms of round-trip time (RTT) for adaptive applications (such as those using TCP). Figure 2.1 and Figure 2.2 show the principle of the one-way delay measurement and the RTT delay measurement respectively. A discussion on the usefulness and weakness of the one-way delay metric and the RTT metric can be found in [8] and [10].

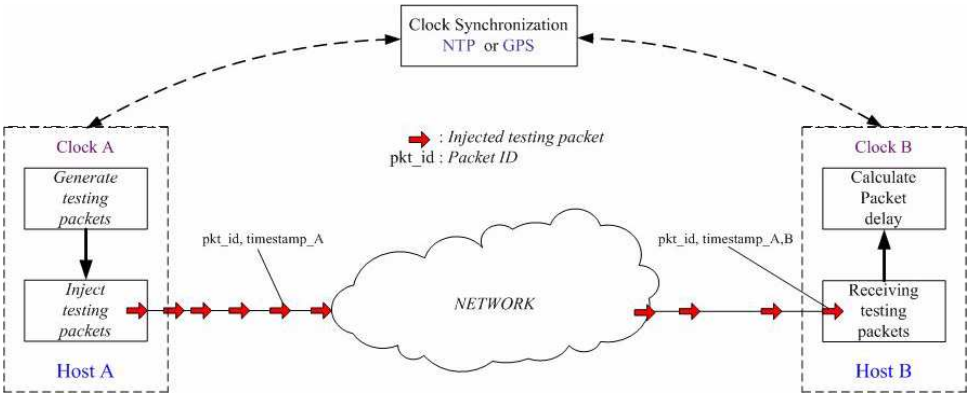


Figure 2.1 An illustration of one-way delay measurement

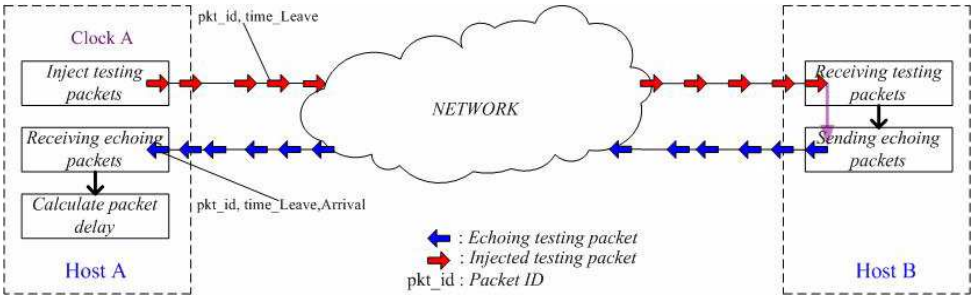


Figure 2.2 An illustration of round-trip delay measurement

2.1.3.1 One-way Delay Measurement

The biggest challenge in one-way delay measurements is clock synchronisation. Simply speaking, host A and host B at both ends of the network path must use the same clock when measuring one-way delay (see Figure 2.3). Assume that the local time at host A is T_A and the local time at host B is T_B. Without proper synchronisation, an error term $T_B - T_A$ will be present in one-way delay measurements, which cannot be easily removed. For one-way delay measurements, it is usually required that $|T_B - T_A| \leq 1$ ms in order to achieve a reasonably accurate measurement.

There are two basic methods for achieving synchronisations: one uses GPS and the other uses network protocols such as Network Time Protocol (NTP) [13].

The use of GPS devices in the monitoring system can dramatically increase cost.

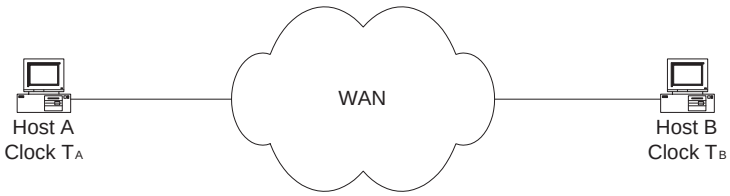


Figure 2.3 Clock Synchronisation

More importantly, since GPS requires line-of-sight between the equipment and the GPS satellites, it may not work indoors, underground, or in the presence of obstructions such as buildings or mountains blocking the direct view to these satellites. Due to the aforementioned reasons, despite the widespread use of GPS in some large-scale performance monitoring projects, we do not consider GPS to be a viable option in this project.

On the other hand, two important concerns arise if the hosts at both ends of the network path derive their time using a network synchronisation protocol such as NTP [13]:

- First, NTP's accuracy depends in part on the properties (particularly delay) of the Internet paths used by the NTP peers, and these are exactly the properties that we wish to measure, so it would be unsound to use NTP to calibrate such measurements.
- Second, NTP focuses on clock accuracy, which can come at the expense of short-term clock skew and drift [13]. For example, when a host's clock is synchronized to a time source (e.g., network time server), if the synchronisation occurs infrequently, then the host will sometimes be faced with the problem of how to adjust its current, incorrect time, T_i, with a considerably different, more accurate time that it has just learned, T_a. Two general ways in which this is carried out are to either immediately set the current time to T_a, or to adjust the local clock's update frequency (hence, its skew) so that at some point in the future the local time T_i' will agree with the more accurate time T_a'. The first mechanism introduces discontinuities and can also violate common assumptions that timestamps are monotone increasing. If the host's clock is set

backward in time, sometimes this can be easily detected. If the clock is set forward in time, this can be harder to detect. The skew induced by the second mechanism can lead to considerable inaccuracies when computing one-way delay.

2.1.3.2 Round-trip Delay Measurement

In comparison with one-way delay measurements, measurement of round-trip delay has two advantages.

- Ease of deployment. Unlike one way delay measurements, RTT is measured by using the clock at the same host, so there is no synchronisation problem in RTT measurements.
- Ease of interpretation. As discussed in the previous paragraph, in some circumstances, the round-trip time is in fact the quantity of interest.

The major problem that needs to be taken care of in RTT measurements is the time spent by the destination host in receiving and recognising the packet from the source, and then producing and sending the corresponding response, which adds an additional error and uncertainty to the RTT measurements. This systematic error needs to be calibrated in the RTT measurements [10].

In this project, we measure RTT instead of one-way delay because Optus specifies in [14] that RTT is the quantity of interest. RTT is also the metric currently used in most SLAs.

2.1.4 Jitter

Jitter, sometimes called delay variation, is the difference between the one-way delay of the selected packets [12]. Generally, jitter is specified as the absolute value of delay difference between selected packets [12], [15]. Despite the fact that jitter is derived from one-way delay measurements, time synchronisation is not a major problem in jitter measurements. Let a_i represent the departure epoch of packet i at

the source host and b_i represent the corresponding arrival epoch at the destination host. Let D_i be the delay experienced by the i-th packet when travelling from the source to the destination, i.e., $D_i = b_i - a_i$. Denote the jitter between the i-th and the $(i-1)$-th packets by j_i, then

$$j_i \equiv D_i - D_{i-1} = (b_i - a_i) - (b_{i-1} - a_{i-1}) = (b_i - b_{i-1}) - (a_i - a_{i-1}). \quad (2.1)$$

The first term $b_i - b_{i-1}$ on the right-hand side of Equation 2.1 is the difference in the arrival epochs of packet i and packet $i-1$ at the destination host, which only needs the local time at the destination host for computation. The second term $a_i - a_{i-1}$ is the difference in the departure epochs of packet i and packet $i-1$ at the source host, which only needs the local time at the source host for computation. The difference operation easily removes any constant error between the source clock and the destination clock when measuring jitter. Provided that the sampling interval between packet i and packet $i-1$ is small, which is generally true, any higher order clock error (e.g., skew) can also be ignored. As such, synchronisation does not affect jitter measurements.

2.1.5 Packet Loss

In certain real-time applications (such as VoIP and mobile video), the loss pattern or loss distribution is a key parameter that determines the performance observed by the users. For the same long-term packet loss rate, different loss patterns lead to different application-level QoS perceived by the users [9], [16], [17], [18], [19]. Also, many forward error recovery approaches become less efficient as the loss burstiness (i.e., number of consecutive packet losses) increases. Thus it is important to not only measure the mean loss rate but also to measure the loss distribution. The loss distribution is customarily parameterised by such metrics as loss burstiness and the distances between loss bursts.

Several models have been proposed in the literature. Before describing these models, we first present an introduction to the key technological components. The loss indicator function Y for a stream of packets is defined as:

$$Y(i) = \begin{cases} 0: & i\texttt{-th packet is successfully received} \\ 1: & i\texttt{-th packet is lost} \end{cases} \quad (2.2)$$

Loss period length [11] is the number of consecutive packet losses, which is bounded by packets that have been successfully delivered. It is also referred to as "loss run length" in the literature. The inter-loss period length is the distance between the last packet considered lost in "loss period" $i-1$ and the first packet considered lost in the "loss period" i, i.e., the number of consecutive packets that have been successfully delivered. It is also referred to as "no-loss run length" or "good run length" in the literature. There are four popular packet loss models in the literature, i.e., the Bernoulli loss model, the two-state Markov chain model, the n-th order Markov chain model and the extended Gilbert model.

2.1.5.1 Bernoulli Loss Model

In the Bernoulli loss model [20], the packet loss is assumed to be independent. That is, the probability of $Y(i)$ being either 0 or 1 is independent of all other values of Y and the probabilities are the same irrespective of i. This model is characterised by a single parameter, r, the probability of $Y(i)$ being 1 (corresponds to a packet loss). Parameter r can be obtained from the measurement as the average packet loss ratio. The inter-loss period length distribution for this model is

$$f(k) = r(1-r)^{k-1} \; for \; k = 1, 2, ..., \infty, \tag{2.3}$$

and the loss period length distribution is

$$f(k) = (1-r)r^{k-1} \; for \; k = 1, 2, ..., \infty. \tag{2.4}$$

2.1.5.2 Two-state Markov Chain Model

This is also known as the Gilbert model [19]. In the Gilbert model, the current state, $Y(i)$ (i.e., whether the current packet is lost) of the stochastic process depends only on the previous value $Y(i-1)$. Unlike the Bernoulli model, this model is able to capture the dependence between consecutive losses. Figure 2.4 illustrates the Gilbert model. The Gilbert model is characterised by two parameters, p and q, which are the transition probabilities between the two states:

$$p = Pr[Y(i) = 1 | Y(i-1) = 0], \;\; q = Pr[Y(i) = 0 | Y(i-1) = 1]. \tag{2.5}$$

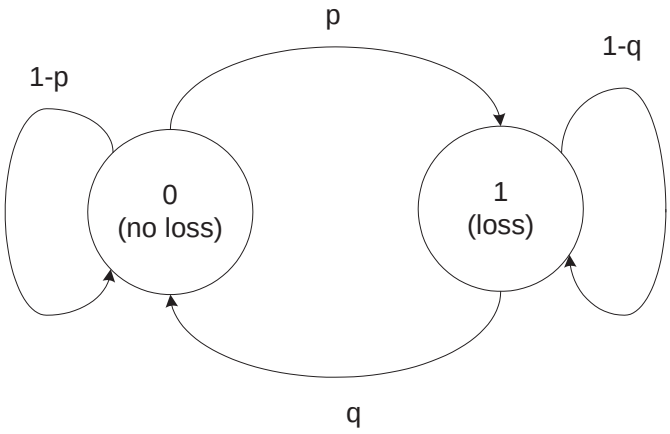


Figure 2.4 The Gilbert Model

The loss period distribution is

$$f(k) = q(1-q)^{k-1} \; for \; k = 1, 2, ..., \infty. \tag{2.6}$$

2.1.5.3 n-th order Markov Chain Model

The Bernoulli model and the Gilbert model are special cases of this class of models. Different from the Bernoulli model, which assumes that packet losses are entirely independent, and the Gilbert model, which assumes that the current packet loss depends only on the previous one-packet event, in the n-th order Markov chain model [20], the current state of the process depends on a certain number of previous packet events which is the order of the process. Such a process is characterised by its order n and by an $n \times n$ conditional probability matrix.

A process $\{Y(i)\}$ is a Markov chain of order n if the value of $Y(i)$ is independent of $Y(m), m < i - n$ and is dependent on $Y(I), i - n < I \leq i - 1$. In reality the value of n can be determined by examining the autocorrelation of $\{Y(i)\}$.

Yajnik *et al.* [20] show that their packet traces typically have $n \leq 6$, and some require n to be 20 to 40. They did not quantify how much precision is gained by using an n-th order Markov model as compared to other simple models such as the 2-state Gilbert model.

2.1.5.4 Extended Gilbert Model

Sanneck *et al.* [19] propose a different model that leads to fewer states, which is often referred to as the extended Gilbert model. Their key distinction is that a general n-th order Markov chain model assumes all past n events can affect the current state (i.e., whether the current packet is lost); whereas in an extended Gilbert mode only the past (up to) n consecutive loss events will affect the current state. Figure 2.5 illustrates the extended Gilbert model. Sanneck *et al.* [19] also provide the equations for computing the parameters of the extended Gilbert model. Theoretically, loss period length can have an infinite value, which implies that the extended Gilbert model may have an infinite number of states. However, in reality, the maximum number of states in the extended Gilbert model is limited by both the maximum loss period length in real measurements and by the applications being considered. A number of packet loss

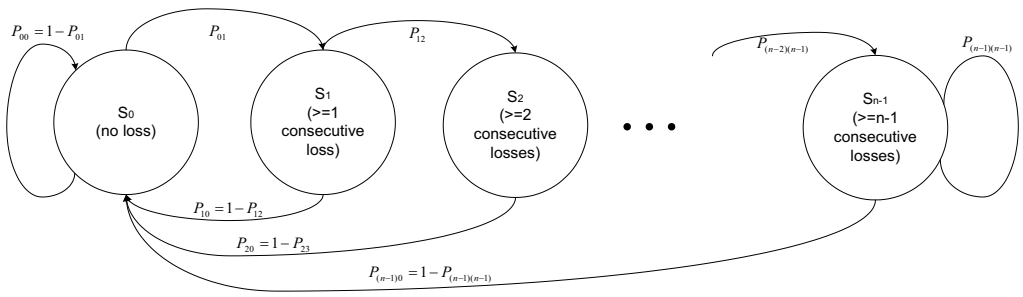


Figure 2.5 The Extended Gilbert Model

measurement studies have shown that the maximum number of consecutive packet losses is typically less than four [20], [21], [22]. Occasionally, this value may exceed 10 for some traffic traces. The analysis in [19] reveals that only a few burst losses larger than 10 packets take place, and thus models with a higher number of states do not provide much additional information.

Moreover, the number of states in the extended Gilbert model is dependent on the network utilisation. In light network utilisation, packet losses are more likely to be independent, whereas in heavy network utilisation, burst losses are more likely to occur [23]. This implies that a greater number of states are required in heavy network utilisation. Whether the end-to-end path contains wireless links may also

affect the model parameters, as burst losses are more likely to occur in the wireless environment.

The extended Gilbert model can be further enhanced by factoring the inter-loss period length into the states of the extended Gilbert model [19]. Although this may bring some improvements, it makes the model very complex. As such, no real implementation of this enhancement has been reported.

Other noteworthy work in the area includes the definition of the noticeable loss rate (NLR) metric by the IETF IPPM working group [11], [18]. Given a threshold distance d, the number of noticeable losses is losses with an inter-loss distance less than or equal to d. Since some applications may be embedded with a forward error control algorithm or loss concealment algorithm, some sporadic packet losses with a large inter-loss distance can be easily corrected and do not affect the user-perceived QoS. The NLR is proposed to reflect the characteristics of these applications. However, this project is intended to be a general QoS measurement project which does not target specific applications. Therefore we do not consider this definition. In [21], a sine model is proposed to model the diurnal behaviour of packet loss. However, the proposed model is too simplistic. Based on the exponential on-off source model, Hasib *et al.* [24] present some analysis of the average time required for probing packets to obtain a valid observation of packet loss. Although their result is not in the form that can be used in real applications, it is an interesting development in the area.

In this project, we shall stay with IETF standards [9], [11] when measuring packet loss and presenting measurement results.

2.2 Network Measurement Methodology

Existing QoS measurement systems can be broadly classified into two categories: passive and active measurements.

2.2.1 Passive Measurement

Passive measurements are used to observe actual traffic without injecting extra traffic into the network. There are two basic methods for obtaining end-to-end QoS parameters in passive measurements. In the first method, equipment similar to a network analyser is used. Two pieces of measurement equipment are deployed at both ends of the network path/segment to be measured. The two measurement equipments are often synchronised by using GPS or NTP [25]. The measurement equipment captures and measures packets passing by it. The equipment keeps a record of both the measurement (e.g., time of arrival) and the packet information (e.g., packet header or a scrambled version of the packet header, which allows unique identification of the packet). End-to-end QoS metrics can then be obtained in offline conditions by comparing measurement information of the same packet captured by both pieces of equipment. Papagiannaki *et al.* use this method to obtain the single-hop delay of a router [26], [27]. In the second method, each network element keeps the statistical information of QoS parameters (e.g., distribution); the end-to-end QoS statistics can be obtained by correlating the statistics of the QoS parameters in each network segment along the path. Data collection in the second method can be achieved by piggybacking data onto existing network protocols (e.g., SNMP). Some papers have reported using this method to obtain end-to-end packet loss from packet loss measurements in each network element along the path. However, this method relies on the assumption that the statistical characteristics of QoS parameters in different network elements along the path are independent, which may not be true. Therefore, the validity of the second method is doubtful.

Typical passive measurement equipments measure such metrics: throughput, utilisation, availability, one-way delay. These metrics only present an overall view of network performance. For more detailed analysis, some other statistics are required, e.g., the packet size distribution, length of packet trains, etc [6].

2.2.2 Active Measurement

Active measurements inject synthetic traffic into the network based on scheduled sampling (by sending probing packets) in order to observe network performance.

The principle is that the structure (packet size distribution and inter-departure time distribution) of the synthetic traffic is known, and so by measuring how it is affected by the network it traverses, network performance can be obtained.

Compared with passive measurements, active measurements have their own advantages and disadvantages. Active measurements are easy to implement. They do not need the cooperation of intermediate nodes along the path. Hence no hardware change is required, which can significantly reduce the cost of measurements. This makes active measurements an attractive option for small-scale performance monitoring. Currently most performance measurement systems are designed on the basis of active measurements. In addition, active measurements are much easier to conduct, repeat and vary as often as desired. The volume of active measurement data is also markedly reduced compared to the passive monitoring of high bandwidth links. However, active measurements are not scalable. Synthetic traffic injected into the network for monitoring a large number of SLA flows may degrade the performance of the network and cause traffic congestions, and the capability of active measurements is often limited by constraints on network capacity consumed by synthetic traffic. Due to their intrusive nature, the accuracy of measurements greatly depends on the sampling frequency and the sampling method and this is a drawback. Obviously, insufficient probing packets make the measurement results unreliable. On the other hand, too many probing packets may cause an extra burden on the network and change the statistical characteristics of the QoS parameters to be observed. The performance of using active packet probing is also highly dependent on the variability of the traffic characteristics. Therefore, a key challenge in designing an active measurement system is to design a statistical sampling strategy matching the statistical characteristics of network traffic in order to obtain the most accurate measurements. Details of the sampling strategy will be discussed in Chapter 5.

Typical active measurement equipments measure such metrics: RTT delay, one-way delay, one-way packet loss, availability, TCP performance, topology discovery, routing dynamics [6].

2.3 Summary

This chapter reviewed related work in the area. Firstly, a brief introduction of the main usages of Internet measurements and standard metrics in network measurements was presented. A detailed discussion of characteristics of QoS metrics related to the monitoring system design was presented. Secondly, two main network measurement methodologies (active and passive measurements) were introduced, including their advantages and disadvantages. We chose an active measurement methodology to monitor the RTT delay, one-way jitter and one-way loss in the monitoring system.

Chapter 3

Sampling Techniques

3.1 Introduction

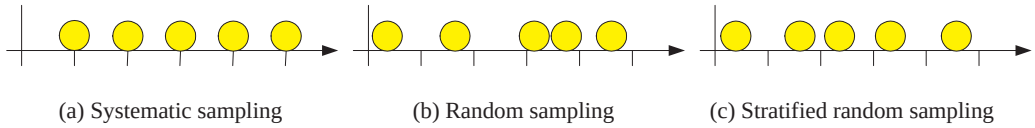
There are two basic approaches for generating synthetic traffic in active measurements. One is traffic modelling and the other is sampling. Traffic modelling attempts to model the behaviour of a specific network application (e.g., VoIP) by generating traffic with similar statistical characteristics to those generated by the application. Since different network applications have different traffic characteristics, the traffic modelling approach is often used to obtain an estimate of QoS experienced by a specific application in the network. For different applications, different traffic models need to be used. In comparison, sampling aims to obtain the characteristics of the parent population (e.g., all packets generated by a source network) at a lower cost by observing only a small subset of the parent population [28] rather than the entire population. In this project, sampling techniques are employed into the performance monitoring system to reduce the amount of control data, and the resources required to process it.

In this chapter, three conventional sampling techniques, i.e., systematic sampling, random sampling and stratified sampling, and their characteristics are introduced. Then a new sampling technique called “adaptive sampling” is presented. The triggering mechanism of the sampling process is also discussed. Finally, the limitation, i.e., the minimum sample size required for a specific accuracy with a given confi-

dence level and an error bound is presented.

3.2 Sampling Techniques

Existing sampling techniques can be classified into three categories: systematic sampling, random sampling and stratified sampling [4], [28], [29]. Figure 3.1 illustrates these three sampling techniques.



(a) Systematic sampling (b) Random sampling (c) Stratified random sampling

Figure 3.1 Sampling techniques

3.2.1 Systematic Sampling

Systematic sampling generates sampling traffic according to a deterministic function. Generation of the sampling traffic is triggered by either time (i.e., at fixed intervals) or packet count (i.e., every N-th packet). Figure 3.1.(a) shows periodic sampling with a period of T seconds.

The use of systematic sampling always involves the risk of biasing the results. If the systematics (e.g., periodic repetition of an event) in the sampling process resemble the systematics in the observed stochastic process (e.g., occurrence of event of interest in the network), there is a high probability that the estimation will be biased. In this context it also has to be considered that there might be systematics in the observed process that one might not be aware of in advance [28]. Typical examples of the systematics in the network are the periodic update of the routing table by a router, which has been shown in the literature to contribute to the periodic surge in packet delay, and the periodic exchange of information between routers due to SNMP protocol.

3.2.2 Random Sampling

Random sampling employs a random distribution function to determine when a sample should be generated. Typically the samples are generated according to a Poisson process. As shown in Figure 3.1.(b), random sampling may produce a varying number of samples in a given time interval. With random sampling, an unbiased estimate of the QoS metric can be achieved [30, p. 21]. However, the entirely random nature of the sampling process may also cause the undesirable effect that sampling intervals are not uniformly distributed, and therefore the network may not be sampled for a rather long time.

3.2.3 Stratified Sampling

Stratified random sampling combines the fixed time interval used in systematic sampling with random sampling [31]. Figure 3.1.(c) shows stratified random sampling with a period of T and a random sample is generated in each period.

Stratified sampling divides the sampling process into multiple steps. Firstly the elements of the parent population are grouped into subsets (i.e., strata) according to a given characteristic. This grouping can be undertaken in multiple steps. Then samples are taken from each subset. Because the selections in different strata are made independently, the variances of estimators for individual strata can be added together to obtain the variance of the estimator for the whole population. A smaller variance indicates a more accurate estimator. Since only the within-stratum variances enter into the variance of the estimator, the principle of stratification is to partition the population in such a way that the units within a stratum are as similar as possible. The stronger the correlation between units within a stratum, the more accurate the estimator will be. Then, even though strata may differ markedly from one another, a stratified sample with the desired number of units from each stratum in the population will tend to be “representative” of the population as a whole [32, pp. 117]. A typical example of stratified sampling is dividing time into fixed intervals according to the correlation of the elements (e.g., delay) to be measured, then generating sampling packets according to a random process during each interval. The stronger the correlation between packet delays in an interval is, the more accurate the mean

delay estimate will be. In [29], Zseby divides the parent population into different strata according to packet size. His method may provide a more accurate mean delay estimate provided that the packet delay has a strong correlation with packet size.

Stratified sampling may reduce the sample size if *a priori* knowledge (e.g., correlation in packet delay) is taken into account for building strata [28]. Depending on how the sample size is distributed among strata, stratified sampling can be further classified into proportional allocation and optimum allocation [32]. **Proportional allocation** means that the sample size in each stratum is proportional to the size of parent population in that stratum, while **optimum allocation** means that the sample size in each stratum is proportional to the standard deviation of the variable of interest in that stratum.

3.2.4 Adaptive Sampling

Another kind of sampling method is adaptive sampling. In conventional sampling techniques (e.g., the three sampling techniques in the earlier paragraphs), the sample selection procedure does not depend on the observations made during the sampling, so that the entire samples may be selected prior to the start of the sampling process. In adaptive sampling, the procedure for selecting samples may depend on the values of the variable of interest observed during the sampling process. The primary purpose of adaptive sampling design is to take advantage of population characteristics to obtain more precise estimates, for a given sample size or cost, than is possible with conventional designs. For example, the dynamic nature of network traffic determines that sometimes the variable of interest (e.g., packet delay, packet loss, traffic quantity) may be smooth, while an other time, the variable of interest may present dramatic variations. Intuitively, given a fixed total sample size, a more accurate estimate can be obtained by changing the sampling rate adaptively such that the algorithm samples less during periods in which the variable of interest is smooth and samples more during periods in which the variable of interest varies dramatically. Figure 3.2 shows the adaptive sampling in two measurement intervals. In the measurement of interval i, the variable of interest presents dramatic fluctuation, so we select comparatively more samples; while in the measurement of interval $i + j$, the variable of interest

changes smoothly, so we select comparatively fewer samples.

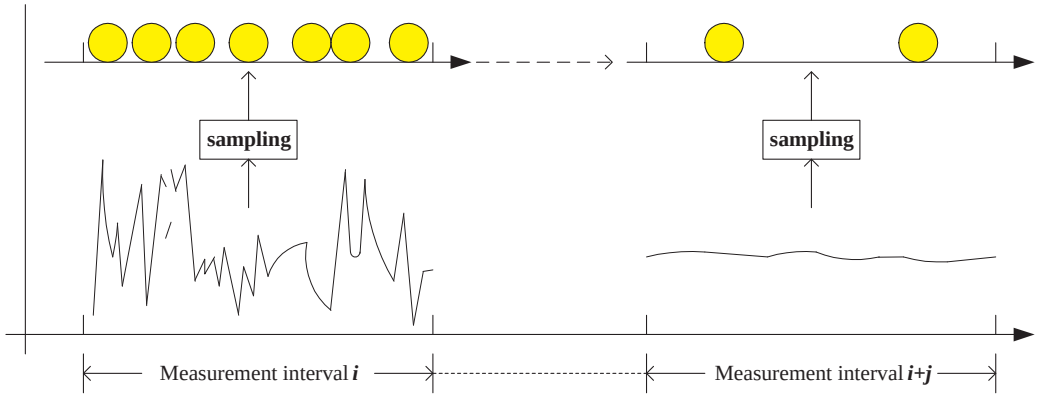


Figure 3.2 Adaptive Sampling

Despite its advantages, the real implementation of adaptive sampling may be difficult, which may compromise its advantages. For example, it is stated in the literature [32, pp. 123] that for stratified sampling, the most accurate estimate is obtained by allocating the number of samples in each stratum so that the number of samples in each stratum is proportional to the standard deviation of the variable of interest in the stratum. Then, to implement adaptive stratified sampling for packet delay measurements, the optimum sampling design should allocate the number of samples in each stratum to be proportional to the standard deviation of packet delay in that stratum. Therefore, to determine the optimum number of samples for the next stratum, the standard deviation of packet delay in the next stratum has to be predicted. In reality, the uncertainty and complexity involved in standard deviation prediction may compromise the advantage of using the adaptive stratified sampling technique.

3.2.5 Sampling Trigger

The sampling process can be triggered by packet count, timer or packet-content [33], [28]. In count-based sampling methods, the start and the finish of a sampling is triggered by packet count. For example, a count-based systematic sampling deterministically selects every k-th element (e.g., packet) out of the data set. Timer-based sampling methods use a timer instead of a packet count to trigger sampling. When

the timer expires, we capture (in passive measurement) the next packet to arrive or inject (in active measurement) a synthetic probe packet into networks. Packet-content-based sampling methods trigger the sampling process according to the contents of a packet (e.g., TCP SYN packet, the value of specified fields in the packet header, the type of packet, etc.).

With count-based sampling, the time interval between sampling packets is variable, while with timer-base sampling, the time interval is constant, but the number of packets between adjacent sampling is variable [33]. Claffy *et al.* prove in [4] that the performance difference between count-based sampling techniques and timer-based sampling techniques is very small. In this project, timer-based sampling methods are employed in the monitoring software.

3.3 Accuracy of Sampling

A fundamental problem that needs to be addressed in sampling design is the accuracy of the estimate obtained through sampling. To motivate the problem, let us consider two simple cases. In case A, a packet loss is observed among ten sampling packets. Therefore, a mean packet loss ratio of 10% is obtained. In case B, ten packet losses are observed among one hundred sampling packets. The same mean packet loss ratio of 10% is obtained. An observer will naturally conclude that the mean packet loss ratio obtained in case B is more accurate than that obtained in case A. According to the **Central Limit Theorem**, given a large sample size, regardless of the statistical characteristics of the parent distribution, the distribution of the mean of samples approaches a Gaussian distribution with a mean equal to the parent population's mean, and a standard deviation equal to the standard deviation of the parent population divided by the square root of the sample size. Therefore, the sample size needs to be larger if a higher confidence level is required. Then a problem arises as to what is the minimum sample size required in order to obtain an estimate satisfying a given accuracy criterion.

In order to model this problem we make the following assumptions and simplifications [32]:

- We assume that the injected packets (samples) do not disturb the network characteristics (e.g., packet loss) which are to be observed;
- The metric of interest is the proportion of packets, p, having a common attribute (e.g., being lost);
- In the sequence of packets, a packet with the attribute of interest is considered as a hit ($y = 1$) and a packet without the attribute is considered as a no-hit ($y = 0$);
- Packets with the attribute of interest occur independently in the sequence;
- The sample size is small compared to the parent population (e.g. total number of packets from a source network) and satisfies the condition:

$$\frac{n}{N} \leq 0.05. \tag{3.1}$$

Condition (3.1) is required in order for the *finite population correction* factor (i.e. $(N - n)/N$) [32, pp. 15] to be approximated by 1.

With these assumptions the network process can be modelled as a discrete time $\{0, 1\}$-valued stochastic process Y. The estimation of p can then be treated as the estimation of the population mean of the stochastic process Y. Table 3.1 shows the parameters used in the subsequent analysis.

Table 3.1 Parameters used in the analysis

Denotation	Meaning
N	Total number of packets in the parent population
n	Number of sampling packets
V	Number of hits in the parent population
v	Number of hits in the sample
p	Real proportion of packets with a given attribute
$\hat{p}$	Estimated proportion of packets with a given attribute through sampling
ε	Absolute estimation error
$1 - \alpha$	Confidence level

The objective of sampling is to produce an estimator

$$\hat{p} = \frac{v}{n} \tag{3.2}$$

that is within ε around the true value

$$p = \frac{V}{N}, \tag{3.3}$$

with a probability greater than or equal to $1 - \alpha$:

$$Pr(|\hat{p} - p| \leq \varepsilon) \geq 1 - \alpha. \tag{3.4}$$

The population mean μ is the average of the y-values in the whole population:

$$\mu = \frac{1}{N} \sum_{i=1}^{N} y_i = \frac{V}{N} = p. \tag{3.5}$$

The sample mean $\overline{y}$ is the average of the y-values in the sample:

$$\overline{y} = \frac{1}{n} \sum_{i=1}^{n} y_i = \hat{p}. \tag{3.6}$$

Also, with simple random sampling, the sample variance s^2 is an unbiased estimator of the *finite population variance* σ^2. The finite population variance is:

$$\sigma^2 = \frac{1}{N-1} \sum_{i=1}^{N} (y_i - \mu)^2 = \frac{1}{N-1} \sum_{i=1}^{N} (y_i^2 - \mu^2). \tag{3.7}$$

Since Y is a $\{0,1\}$-valued stochastic process, $y_i^2 = y_i$. Thus

$$\sigma^2 = \frac{1}{N-1} \sum_{i=1}^{N} (y_i - \mu^2) = \frac{N}{N-1} (\mu - \mu^2) = \frac{N}{N-1} p(1-p). \tag{3.8}$$

Similarly, the sampling variance is:

$$s^2 = \frac{1}{n-1} \sum_{i=1}^{n} (y_i - \overline{y})^2 = \frac{n}{n-1} (\overline{y} - \overline{y}^2) = \frac{n}{n-1} \hat{p}(1-\hat{p}). \tag{3.9}$$

The variance of the estimator $\overline{y}$ with simple random sampling is [32, pp. 15]:

$$var(\overline{y}) = \frac{N-n}{N} \times \frac{\sigma^2}{n}. \tag{3.10}$$

An unbiased estimator of this variance is:

$$\widehat{var}(\overline{y}) = \frac{N-n}{N} \times \frac{s^2}{n}. \tag{3.11}$$

The quantity $(N-n)/N$ is called the *finite population correction* (fpc) factor. Its value is very close to 1 under condition (3.1). Therefore it can be ignored and

$$\widehat{var}(\overline{y}) \approx \frac{s^2}{n} = \frac{\hat{p}(1-\hat{p})}{n-1}. \tag{3.12}$$

The estimator $\overline{y}$ is a random variable with mean $\mu = p$ and having a binomial distribution. When the sample size is large enough, this binomial distribution can be approximated by a Gaussian distribution. Therefore, from the sampling requirement (3.4), it can be obtained that:

$$\varepsilon = z_{\frac{\alpha}{2}} \sqrt{\widehat{var}(\overline{y})} = z_{\frac{\alpha}{2}} \times \sqrt{\frac{\hat{p}(1-\hat{p})}{n-1}}, \tag{3.13}$$

where $z_{\frac{\alpha}{2}}$ is the upper $\frac{\alpha}{2}$ quantile of the normal distribution. From Equation 3.13, the minimum sample size required to satisfy condition (3.4) can be obtained:

$$n \geq \frac{z_{\frac{\alpha}{2}}^2 \times \hat{p}(1-\hat{p})}{\varepsilon^2} + 1. \tag{3.14}$$

Fewer samples will be required for a smaller value of $\hat{p}$ in order to maintain the absolute error. The maximum value of $\frac{z_{\frac{\alpha}{2}}^2 \times \hat{p}(1-\hat{p})}{\varepsilon^2}$ is obtained when $p = 0.5$.

Based on the earlier analysis, given a specific performance target (e.g., the system needs to observe a packet loss ratio as small as p with an accuracy of ε), the minimum number of samples required can be computed. Alternatively, given the number of samples, the accuracy of the estimate can be obtained.

For the estimation of the mean packet delay, let μ denote the true mean packet delay of the parent population, and $\hat{\mu}$ denote an estimate of the true mean packet delay μ. In the same way, we can obtain the minimum sample size n for estimating the mean packet delay with a given confidence level $1-\alpha$:

$$n \geq \frac{z_{\frac{\alpha}{2}}^2 \sigma^2}{r^2 \mu^2}, \tag{3.15}$$

or with s^2 substituting σ^2 and $\hat{\mu}$ substituting μ,

$$n \geq \frac{z_{\frac{\alpha}{2}}^2 s^2}{r^2 \hat{\mu}^2}. \tag{3.16}$$

Here r represents the bounds of the relative error between the actual value and its estimate.

3.4 Summary

In this chapter, I described major sampling techniques that can be used in the sampling-based monitoring system, which include three conventional sampling techniques and a new sampling technique called “adaptive sampling”. Then I presented a theoretical analysis of the minimum sample size required in order to obtain an estimate satisfying a given accuracy criterion.

Chapter 4

Performance Comparison of Different Sampling Techniques

4.1 Introduction

In this chapter, I compare the performance of different sampling techniques. As variance of the sample mean has been widely used as a performance measure [30, pp. 15], [34], the performance of these sampling techniques is compared by comparing the variance of the sample mean of different sampling schemes under the constraint that the sample sizes of different sampling methods are the same. The smaller the variance is, the better performance the sampling technique has. The sampling gain Δ is defined as the difference between the variance of the sample mean of two different sampling techniques [29].

I start with a comparison between systematic sampling and random sampling. With count-based sampling, systematic sampling is theoretically more accurate than simple random sampling if the average value of the variances among all possible systematic samples is larger than the variance of the parent population. With timer-based sampling, the performance difference between these two sampling methods is presented in the form of the autocorrelation ρ of the parent population. Comparisons between simple random sampling and stratified sampling with proportional allocation and stratified sampling with optimum allocation are then presented. Simulation results are also displayed to validate the theoretical analysis.

The following notations listed in Table 4.1 are used in our analysis:

Table 4.1 Notations used in the analysis

Denotation	Meaning
N	Total number of packets in the parent population (i.e. parent size)
n	Total number of sampling packets (i.e. sample size)
N_l	Size of the parent population in the l-th stratum
n_l	Sample size in the l-th stratum
L	Total number of strata
K	Maximum number of samples by systematic sampling (n out of N)
μ	Parent population mean
$\bar{y}$	Sample mean
μ_l	Parent population mean in the l-th stratum
$\bar{y}_l$	Sample mean in the l-th stratum
σ_l^2	Variance of the variable of interest in the l-th stratum
y	Variable of interest (e.g. packet delay)

The parent population mean in the l-th stratum, the sample mean in the l-th stratum and the variance of the parent population in the l-th stratum are given by Equation 4.1, 4.2 and 4.3 respectively:

$$\mu_l = \frac{1}{N_l} \sum_{i=1}^{N_l} y_{li}, \tag{4.1}$$

$$\bar{y}_l = \frac{1}{n_l} \sum_{i=1}^{n_l} y_{li}, \tag{4.2}$$

$$\sigma_l^2 = \frac{1}{N_l - 1} \sum_{i=1}^{N_l} (y_{li} - \mu_l)^2. \tag{4.3}$$

The following assumptions are used in the analysis. First, it is assumed that the parent population size in each stratum is large enough so that:

$$N_l - 1 \approx N_l. \tag{4.4}$$

It is also assumed that sample size is small in relation to the parent population size, i.e.,

$$\frac{n}{N} < 0.05. \tag{4.5}$$

Equations 4.4 and 4.5 are widely used assumptions in the area [28], [29].

4.2 Delay Traffic Trace

In order to establish the performance of sampling techniques, experiments are necessary. In this thesis, all experiments are performed using a one-way delay trace as the parent traffic trace. This delay trace is generated by importing a real traffic trace into Opnet Modeler. This real traffic trace ("20010613-060000-e1.gz") was collected by the WAND research group at the University of Waikato Computer Science Department. It was captured between 6.00 a.m. and 8.54 a.m. on June 13th, 2001 on a 100Mbps Ethernet link. IP headers in the traffic trace are GPS synchronised and have a time accuracy of $1\ \mu s$. More information on the traffic trace and the measurement infrastructure can be found on the research group's website [35].

The network topology used in the Opnet Modeler is shown in Figure 4.1. The selection of network nodes (e.g., switch, router, link) and background traffic utilisations of the links are shown in Table 4.2. The background traffic utilisations of the links are chosen as shown in Table 4.2 in order to cause the mean packet delay output by the Opnet Modeler to approach the value [36] measured by the WAND research group.

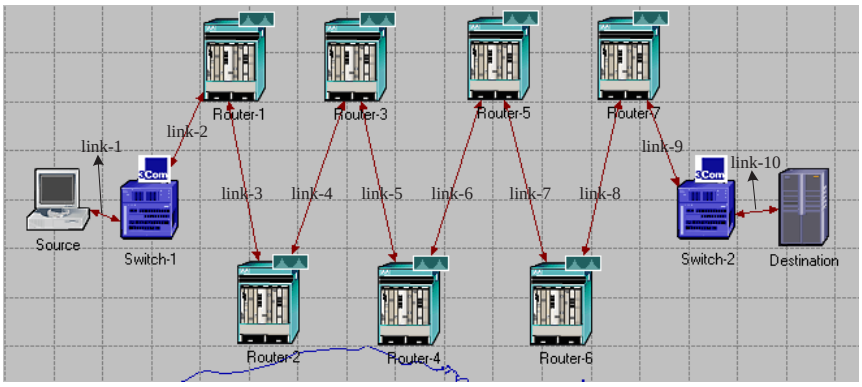


Figure 4.1 Network topology used in Opnet Modeler.

Because a very long time is required to run the simulation, only the first 2600-second part of the entire trace is imported into the Opnet Modeler. After this simulation, I obtain a one-way delay traffic trace with a duration of 2600 seconds. For the purpose of my study, I treat the 2600-second delay traffic trace as the parent population traffic trace. Table 4.3 shows the summary statistics for the packet delay, packet size and

Table 4.2 Selection of network nodes and background traffic utilisations of links

Nodes	Description	Background traffic utilisation
Switch-1,2	$3Com's\ SuperStack\ II\ Switch\ 3800$	N/A
Router-1,2,..,7	$CISCO\ 12008$	N/A
Link-1,10	$100Mbps\ Link$	0%
Link-2,3,8,9	$100Mbps\ Link$	50%
Link-4,7	$100Mbps\ Link$	70%
Link-5,6	$100Mbps\ Link$	55%

inter-arrival time of the parent traffic trace. Figure 4.2 shows the packet inter-arrival time and packet delay of this parent traffic trace.

Table 4.3
Summary statistics for packet delay, packet size and inter-arrival time of the parent delay trace

Property	Min.	Max.	Mean	Median	Var.
Packet delay (ms)	41.092	141.305	86.024	45.2	8529
Packet size (bytes)	24	1478	400.5	34	302080
Inter-arrival Time (ms)	0.006	203.3280	4.5181	0.9880	74.4127

If an accuracy of $r = \pm 5\%$ and a confidence level of $100(1-\alpha)\% = 95\%$ are required in estimating the mean packet delay, then $z_{\frac{\alpha}{2}} = 1.96$ in Equation 3.15. Based on this parent delay trace (total packet number: 577718, mean packet delay: $\mu = 86.024$ ms and the variance of packet delay: $\sigma^2 = 8529$), the minimum sample size is obtained: 1739. Since the parent trace duration is 2600 seconds and the sampling frequency is chosen to be $1\ packet/second$, the actual sample size is approximately 2600, which satisfies the accuracy requirement. Moveover, the ratio between the sample size (2600) and the parent population size (577718) is 0.45%, which complies with the assumption in Equation 4.5. If an accuracy of $r = \pm 1\%$ and a confidence level of $100(1-\alpha)\% = 95\%$ are used, the minimum sample size would be 43464, which means that a much higher sampling rate would be required.

It has been shown that packet size has a significant impact on delay measurements [29], [37], [38] and this impact is to a very large extent independent of the sampling techniques. The focus of this chapter is to evaluate the performance of different

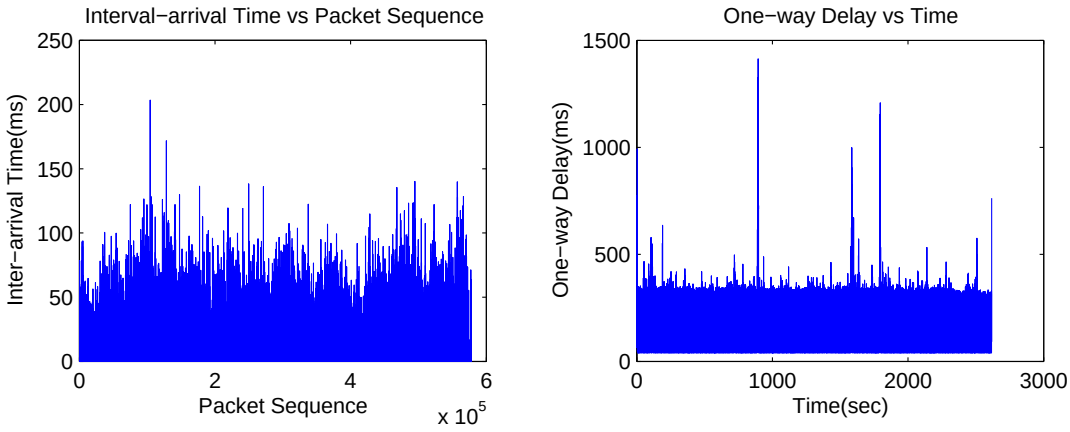


Figure 4.2 Inter-arrival time and packet delay of Parent population traffic trace. Duration: 2600 seconds, packet number: 577718.

sampling techniques. Therefore, to remove the effect of packet size on sampling accuracy, I select the sample delay trace directly from the parent delay trace instead of obtaining it using active sampling by Opnet simulation. The impact of packet size will be discussed in Chapter 4.5.

4.3 Systematic Sampling *vs* Random Sampling

The performance of systematic sampling in relation to that of simple random sampling is greatly dependent on the statistical characteristics (e.g., autocorrelation) of the parent population. There are some parent populations for which systematic sampling is more accurate and others for which it is less accurate than simple random sampling [30, pp. 213]. Therefore, understanding the statistical characteristics of network traffic is critical in order to appropriately estimate network performance. Using the one-way delay traffic trace, I discuss the performance of count-based systematic sampling, count-based simple random sampling, timer-based systematic sampling and timer-based Poisson sampling in the following two subsections.

4.3.1 Comparison between Count-based Systematic Sampling and Count-based Simple Random Sampling

Supposing that the parent population size N is an integer multiple of the sample size n, the maximum number of systematic samples K is computed by $K = N/n$. Then the variance of the sample mean for systematic sampling is:

$$Var_{sys}(\bar{y}) = E(\bar{y} - \mu)^2 = \frac{1}{K}\sum_{k=1}^{K}(\bar{y}_k - \mu)^2, \tag{4.6}$$

where $\bar{y}_k$ is the mean value of the k-th sample in the total K systematic samples:

$$\bar{y}_k = \frac{1}{n}\sum_{i=1}^{n} y_{ki}. \tag{4.7}$$

According to the characteristics of systematic sampling and the assumption $N = nK$, we can obtain that the total K samples are nonoverlapping, and together comprise the entire parent population. Figure 4.3 shows the K systematic samples in the entire parent population. Therefore, the variance of parent population σ^2 can be

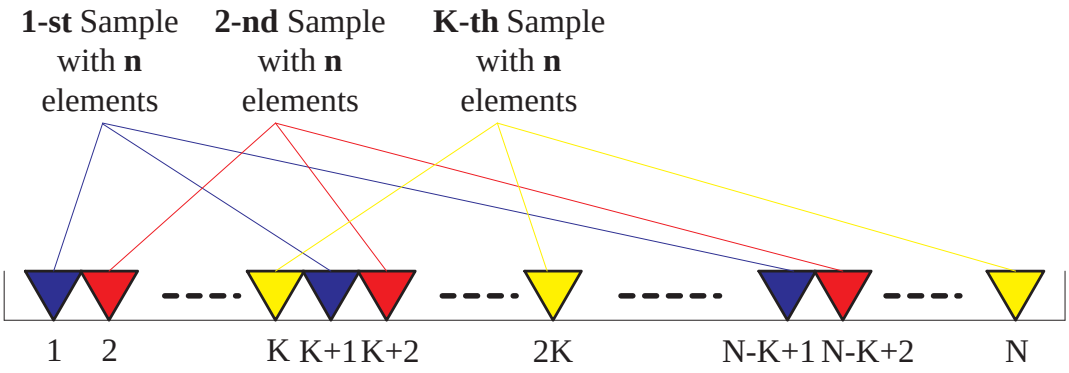


Figure 4.3 K systematic samples in the whole parent population.

written as:

$$\sigma^2 = \frac{1}{N-1}\sum_{j=1}^{N}(y_j - \mu)^2 = \frac{1}{N-1}\sum_{k=1}^{K}\sum_{i=1}^{n}(y_{ki} - \mu)^2, \tag{4.8}$$

$$= \frac{1}{N-1}\sum_{k=1}^{K}\sum_{i=1}^{n}(y_{ki} - \bar{y}_k + \bar{y}_k - \mu)^2 \tag{4.9}$$

$$= \frac{1}{N-1}\sum_{k=1}^{K}\sum_{i=1}^{n}(\bar{y}_k - \mu)^2 + \frac{1}{N-1}\sum_{k=1}^{K}\sum_{i=1}^{n}(y_{ki} - \bar{y}_k)^2, \tag{4.10}$$

$$= \frac{nK}{N-1}\frac{1}{K}\sum_{k=1}^{K}(\bar{y}_k - \mu)^2 + \frac{K(n-1)}{N-1}\frac{1}{K}\sum_{k=1}^{K}\frac{1}{n-1}\sum_{i=1}^{n}(y_{ki} - \bar{y}_k)^2, \quad (4.11)$$

$$= \frac{nK}{N-1}Var_{sys}(\bar{y}) + \frac{K(n-1)}{N-1}\bar{S^2}. \quad (4.12)$$

where

$$\bar{S^2} = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{n-1}\sum_{i=1}^{n}(y_{ki} - \bar{y}_k)^2, \quad (4.13)$$

is the mean value of the sample variances of all K systematic samples. Hence,

$$Var_{sys}(\bar{y}) = \frac{N-1}{N}\sigma^2 - \frac{N-K}{N}\bar{S^2} = \frac{N-1}{N}\sigma^2 - \frac{n-1}{n}\bar{S^2}. \quad (4.14)$$

From Equation 3.10,

$$Var_{ran}(\bar{y}) = (1 - \frac{n}{N})\frac{\sigma^2}{n}. \quad (4.15)$$

From Equation 4.14 and 4.15, the sampling gain of count-based simple random sampling in comparison with count-based systematic sampling is:

$$\Delta_{ran} = Var_{sys}(\bar{y}) - Var_{ran}(\bar{y}), \quad (4.16)$$

$$= \frac{N-1}{N}\sigma^2 - \frac{n-1}{n}\bar{S^2} - (1 - \frac{n}{N})\frac{\sigma^2}{n}, \quad (4.17)$$

$$= (\frac{N-1}{N} - \frac{N-n}{nN})\sigma^2 - \frac{n-1}{n}\bar{S^2}, \quad (4.18)$$

$$= \frac{n-1}{n}(\sigma^2 - \bar{S^2}). \quad (4.19)$$

The sampling gain Δ_{ran} is positive if $\sigma^2 > \bar{S^2}$, and the converse.

4.3.1.1 Simulation Results

Before I discuss the simulation result, an introduction to the simulation setup and the metrics used for performance comparison in this thesis is presented. As discussed earlier, the sample delay traces are selected directly from the parent delay trace. The sampling goal is to estimate the mean packet delay μ and the variance of packet delay σ^2 of the parent delay trace. Several C programs were developed for sampling the sample delay traces and calculating the estimated mean packet delay $\hat{\mu}$ and the estimated variance of packet delay $\hat{\sigma}^2 = s^2$ from the sample delay traces, where $\hat{\mu}$ is the mean packet delay of the sample delay trace and s^2 is the variance of packet

delay of the sample delay trace. The C programs were also used to obtain other results (e.g., absolute error of the estimated mean) required for performance comparison. For simulation, each kind of sampling (e.g., count-based systematic sampling, timer-based systematic sampling, stratified sampling with optimum allocation, etc.) is repeated a number of times, and the random seed in the C programs is updated every repetition. Let M denote the number of repetitions (i.e., sampling rounds). So after M sampling rounds, we obtain M different sample delay traces. The estimated mean delay $\hat{\mu}$ and estimated variance of delay s^2 are calculated for each sample delay trace in the M sampling rounds. Then we can obtain M estimated mean delay, i.e., $\hat{\mu}_1, \hat{\mu}_2, ..., \hat{\mu}_M$ and M estimated variance of delay, i.e., $s_1^2, s_2^2, ..., s_M^2$. The absolute error of the estimated mean, i.e., $|\hat{\mu} - \mu|$, and the absolute error of the estimated variance, i.e., $|s^2 - \sigma^2|$ are also calculated for the M sampling rounds, where the true values μ and σ^2 are obtained in Section 4.2 and shown in Table 4.3.

To compare the performance of different sampling methods, several metrics are used, which are:

- Average value of sample mean ($AMean$): the average value of the sample mean of the M sample delay traces. $AMean = \frac{1}{M}\sum_{i=1}^{M}\hat{\mu}_i$, where M is the sampling rounds, $\hat{\mu}_i$ is the mean value of the i-th sample delay trace in the M sample delay traces. The smaller the difference between $AMean$ and μ is, the better the performance is.

- Average sample variance ($AVar$): the average value of the sample variance among the whole M sample delay traces. $AVar = \frac{1}{M}\sum_{i=1}^{M}s_i^2$, where s_i^2 is the variance of the i-th sample delay trace. The smaller the difference between $AVar$ and σ^2 is, the better the performance is.

- Mean square error (MSE): $MSE = \frac{1}{M}\sum_{i=1}^{M}(\hat{\mu}_i - \mu)^2$. The smaller the MSE is, the higher the accuracy is.

- Absolute error of estimated mean ($AEMean$): $|\hat{\mu} - \mu|$, the smaller $|\hat{\mu} - \mu|$ is, the lower the variance of the sample mean $Var(\hat{\mu})$ is.

- Absolute error of estimated variance ($AEVar$): $|s^2 - \sigma^2|$, the smaller $|s^2 - \sigma^2|$

is, the better can s^2 estimate the true variance σ^2.

For count-based systematic sampling, the sample size $n = 2600$ and the parent size $N = 577718$, so the maximum number of systematic sample delay traces $K = N/n = 222$. Simulation for count-based systematic sampling is performed for 222 sampling rounds, and the related statistics (e.g., $\hat{\mu}_i$, s_i^2) are calculated for each sampling round. The mean value $\bar{S^2}$ of the sample variances for the entire 222 samples is also calculated from Equation 4.13: $\bar{S^2} = 8613$. Since the true variance of delay σ^2 of the parent delay trace shown in Table 4.3 is 8529, we have $\sigma^2 < \bar{S^2}$. From the theoretical analysis in Equation 4.19, the sampling gain $\Delta_{ran} < 0$, which means that the count-based systematic sampling should perform better than the count-based random sampling, and the difference of the variance of the sample mean should be $Var_{ran}(\bar{y}) - Var_{sys}(\bar{y}) = \frac{n-1}{n}(\bar{S^2} - \sigma^2) \approx 83$.

Table 4.4 shows the average value of sample mean $AMean$, average value of sample variance $AVar$ and the MSE. We can see that the $AMean$ of the count-based systematic sampling (i.e., 85.879 ms) is closer to the true mean delay $\mu = 86.824ms$ than that of the count-based simple random sampling (i.e., 87.720 ms); and the $AVar$ of the count-based systematic sampling (i.e., 8613) is closer to the true variance of delay $\sigma^2 = 8529$ than that of the count-based simple random sampling (i.e., 8648). We can also see that the MSE of the count-based systematic sampling (i.e., 45) is smaller than that of the count-based simple random sampling (i.e., 92). Therefore, the count-based systematic sampling indeed performs better than the count-based simple random sampling. The difference between their MSE is $92 - 45 = 47$. Figure 4.4 displays the absolute error of the estimated mean and Figure 4.5 displays the absolute error of the estimated variance for the 222 sampling rounds. We can see that the absolute error of the estimated mean $AEMean$ and the absolute error of the estimated variance $AEVar$ between the count-based systematic sampling and the count-based simple random sampling are approximately the same.

Based on the simulation results, a conclusion can be obtained that the count-based systematic sampling achieves better performance than the count-based simple random sampling, but the performance improvement is marginal.

Table 4.4
Main simulation results of count-based systematic sampling and count-based simple random sampling (true values are: $\mu = 86.824$ ms, $\sigma^2 = 8529$).

Sampling method	M	$AMean = \frac{1}{M}\sum \hat{\mu}_i$	$AVar = \frac{1}{M}\sum s_i^2$	$MSE = \frac{1}{M}\sum(\hat{\mu}_i - \mu)^2$
Systematic	222	85.879 ms	8613	45
Simple Random	222	87.720 ms	8648	92

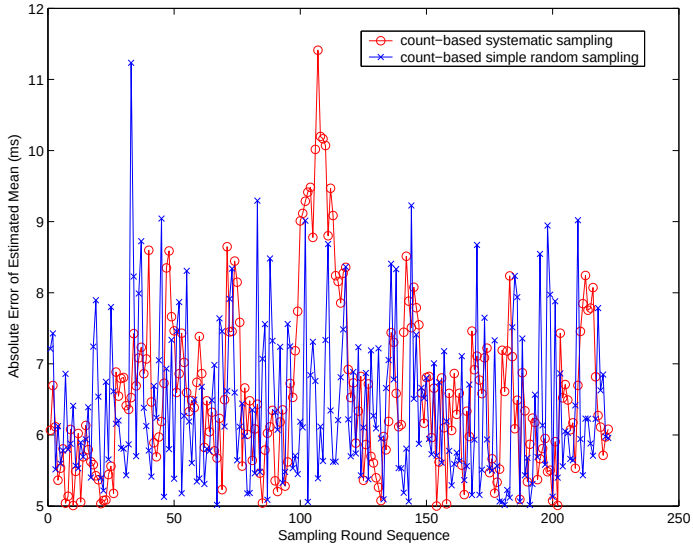


Figure 4.4 Comparison of Absolute Error of Estimated Mean between count-based systematic sampling and count-based simple random sampling. Sample size: 2600, sampling rounds: 222.

4.3.2 Comparison between Timer-based Systematic Sampling and Timer-based Poisson Sampling

For timer-based systematic sampling, the inter-arrival time follows a deterministic function; for timer-based Poisson sampling, the inter-arrival time follows an exponential distribution. In order to compare the performance of the two sampling methods, their variances of the sample mean should be expressed in the same form. The variance of the sample mean of packet delay can be expressed in the form of the autocorrelation of packet delay ρ of the parent delay trace:

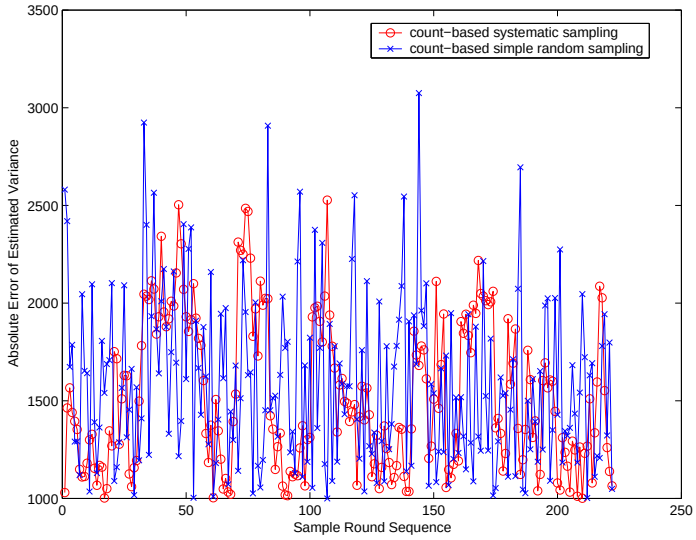


Figure 4.5 Comparison of Absolute Error of Estimated Variance between count-based systematic sampling and count-based simple random sampling. Sample size: 2600, sampling rounds: 222.

$$
\begin{aligned}
Var(\bar{y}) &= E(\bar{y}-\mu)^2 = E(\frac{1}{n}\sum_{i=1}^{n} y_i - \mu)^2, && (4.20)\\
&= E(\frac{1}{n^2}\sum_{i=1}^{n}(y_i-\mu)^2 + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}(y_i-\mu)(y_j-\mu)), && (4.21)\\
&= \frac{1}{n^2}\sum_{i=1}^{n}E(y_i-\mu)^2 + \frac{1}{n^2}E(\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}(y_i-\mu)(y_j-\mu)), && (4.22)\\
&= \frac{1}{n^2}\sum_{i=1}^{n}\sigma^2 + \frac{1}{n^2}E(\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}(y_i-\mu)(y_j-\mu)), && (4.23)\\
&= \frac{\sigma^2}{n}[1+\frac{1}{n\sigma^2}2\sum_{h=1}^{n-1}\sum_{i=1}^{n-h}E((y_i-\mu)(y_{i+h}-\mu))], && (4.24)\\
&= \frac{\sigma^2}{n}[1+2\sum_{h=1}^{n-1}\frac{1}{n}\sum_{i=1}^{n-h}\frac{E((y_i-\mu)(y_{i+h}-\mu))}{E(y_i-\mu)^2}], && (4.25)\\
&= \frac{\sigma^2}{n}(1+2\sum_{h=1}^{n-1}\frac{n-h}{n}\rho(h)), && (4.26)
\end{aligned}
$$

where $\rho(h)$ is the autocorrelation between the i-th and $(i+h)$-th elements in the sample delay trace [39, pp. 16]:

$$\rho(h) = \frac{E((y_i-\mu)(y_{i+h}-\mu))}{E(y_i-\mu)^2}. \quad (4.27)$$

Hence,

$$Var_{sys}(\bar{y}) - Var_{poi}(\bar{y}) = \frac{2\sigma^2}{n} \sum_{h=1}^{n-1} \frac{n-h}{n} (\rho_{sys}(h) - \rho_{poi}(h)), \tag{4.28}$$

where $\rho_{sys}(h)$ is the autocorrelation of delay of the systematic sample trace at lag h, $\rho_{poi}(h)$ is the autocorrelation of delay of the Poisson sample trace at lag h.

Based on Equation 4.28, the comparison between $Var_{sys}(\bar{y})$ and $Var_{poi}(\bar{y})$ becomes the comparison between $\rho_{sys}(h)$ and $\rho_{poi}(h)$. An analytic expression of ρ is therefore desirable for the comparison.

In this thesis, I use an exponential function to approximate the autocorrelation ρ:

$$\rho(t) \approx ae^{-bt} \quad t > 0. \tag{4.29}$$

where a and b are constants.

From Equation 4.29, we have

$$\ln \rho(t) = \ln a + (-b)t. \tag{4.30}$$

Then, the single variable linear regression algorithm can be used to estimate the coefficients a and b. For two groups of random variables $(x_1, x_{2,...,x_n})$ and $(y_1, y_2, ..., y_n)$ related through the linear equation $y_i = kx_i + c + \varepsilon_i$, where k and c are constant coefficients and ε_i is a random variable having a Gaussian distribution with a zero mean, the estimators for the coefficients are [40, pp. 282]:

$$\hat{k} = \frac{n \sum_{i=1}^{n} x_i y_i - (\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} y_i)}{n \sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}, \tag{4.31}$$

$$\hat{c} = \bar{y} - \hat{k}\bar{x}, \tag{4.32}$$

where

$$\bar{y} = \sum_{i=1}^{n} y_i, \tag{4.33}$$

$$\bar{x} = \sum_{i=1}^{n} x_i. \tag{4.34}$$

Hence, let $\ln \rho(t)$ represent y and t represent x, the estimated coefficients are $\hat{b} = -\hat{k} = 0.086743$ and $\hat{a} = e^{\hat{c}} = 0.369142$. Figure 4.6 shows the autocorrelation of

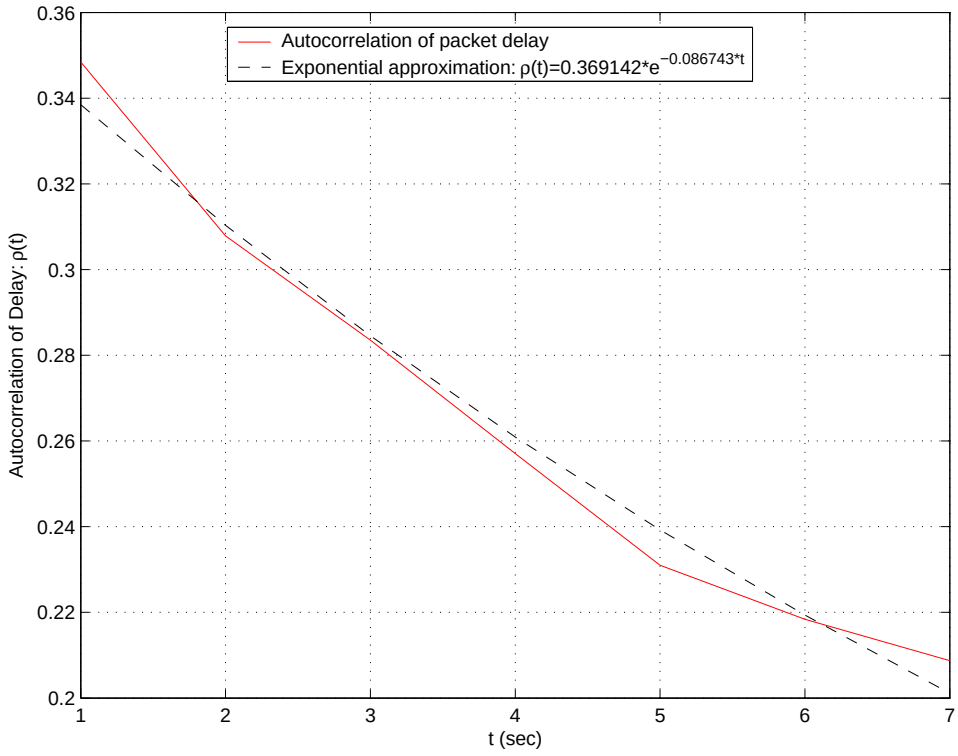


Figure 4.6 Autocorrelation of packet delay and the corresponding exponential approximation (without point $t = 0$).

packet delay of the parent delay trace and the corresponding exponential approximation. We can see that the approximation is appropriate.

For systematic sampling: The adjacent elements in the sample trace are separated by a uniform distance in time, let T_0 denote the uniform distance. Therefore, the autocorrelation between the i-th and the $(i+h)$-th elements in the systematic sample trace is:

$$\rho_{sys}(h) = ae^{-bhT_0}. \tag{4.35}$$

For Poisson sampling: The adjacent elements in the sample trace are separated by a non-uniform distance in time and the inter-arrival time between adjacent elements has a exponential distribution. Hence, the probability density function (PDF) of the inter-arrival time is:

$$p(t) = \lambda e^{-\lambda t} \quad t \geq 0, \lambda \geq 0. \tag{4.36}$$

The mean distance between adjacent elements of the Poisson sample trace is specified as T_0 so that the sampling rate of the Poisson sampling is the same as the sampling rate of the systematic sampling. Hence, the mean inter-arrival time is:

$$E(t) = \frac{1}{\lambda} = T_0. \tag{4.37}$$

Therefore, $\rho_{poi}(1)$, the autocorrelation between adjacent two elements in the Poisson sample trace can be computed as:

$$\rho_{poi}(1) = \int_0^\infty ae^{-bt}\lambda e^{-\lambda t}dt = a\lambda \int_0^\infty e^{-(b+\lambda)t}dt = \frac{a\lambda}{b+\lambda}. \tag{4.38}$$

In order to calculate $\rho_{poi}(2)$,$\rho_{poi}(3)$,..., we need to know the PDF of the sum of two consecutive inter-arrival time slots (e.g., $p_2(t)$), PDF of the sum of three consecutive inter-arrival time slots (e.g., $p_3(t)$), and so on. Let X and Y denote the two consecutive inter-arrival time slots respectively; let T denote the sum of the two consecutive inter-arrival time slots (i.e., $X + Y$); and let $F_2(t)$ denote the distribution function (DF) of T. According to the characteristics of the Poisson process, the two consecutive inter-arrival time slots (i.e., X and Y) are independently identically distributed (i.i.d). Then,

$$F_2(t) = P\{T \le t\} = P\{X + Y \le t\} = \int_0^t \int_0^{t-x} p(x,y)dxdy, \tag{4.39}$$

where

$$p(x,y) = p(x)p(y) = \lambda e^{-\lambda x}\lambda e^{-\lambda y}. \tag{4.40}$$

Hence,

$$\begin{aligned} F_2(t) &= \int_0^t \int_0^{t-x} \lambda e^{-\lambda x}\lambda e^{-\lambda y}dxdy, && (4.41)\\ &= \int_0^t \lambda e^{-\lambda x}dx \int_0^{t-x} \lambda e^{-\lambda y}dy, && (4.42)\\ &= 1 - e^{-\lambda t} - \lambda t e^{-\lambda t}. && (4.43) \end{aligned}$$

The PDF is the derivative of its DF; $p_2(t)$ can be obtained:

$$p_2(t) = F_2'(t) = \lambda^2 t e^{-\lambda t}. \tag{4.44}$$

Therefore, the autocorrelation $\rho_{poi}(2)$ can be calculated as:

$$\rho_{poi}(2) = \int_0^\infty ae^{-bt}\lambda^2 te^{-\lambda t}dt = \frac{a\lambda^2}{(b+\lambda)^2}. \tag{4.45}$$

In the same way, we can obtain $p_3(t)$, $p_4(t)$, $p_4(t)$:

$$p_3(t) = \frac{1}{2}\lambda^3 t^2 e^{-\lambda t}, \tag{4.46}$$

$$p_4(t) = \frac{1}{6}\lambda^4 t^3 e^{-\lambda t}, \tag{4.47}$$

$$p_5(t) = \frac{1}{24}\lambda^5 t^4 e^{-\lambda t}. \tag{4.48}$$

By mathematical induction we can obtain the expression of $p_m(t)$, the details of the derivation are provided in Appendix A.1.

$$p_m(t) = \frac{1}{(m-1)!}\lambda^m t^{m-1} e^{-\lambda t}. \tag{4.49}$$

Finally, the autocorrelation between the i-th and the $(i+h)$-th elements in the Poisson sample trace is obtained from Equation 4.49:

$$\rho_{poi}(h) = \int_0^\infty ae^{-bt}\frac{1}{(h-1)!}\lambda^h t^{h-1}e^{-\lambda t}dt, \tag{4.50}$$

$$= \frac{a\lambda^h}{(h-1)!}\int_0^\infty t^{h-1}e^{-(b+\lambda)t}dt, \tag{4.51}$$

$$= \frac{a\lambda^h}{(h-1)!}\frac{(h-1)!}{(b+\lambda)^{h-1}}\int_0^\infty e^{-(b+\lambda)t}d = \frac{a\lambda^h}{(b+\lambda)^h}. \tag{4.52}$$

From Equation 4.28, 4.35 and 4.52, we can prove the following relationships between $Var_{sys}(\bar{y})$ and $Var_{poi}(\bar{y})$. The details of the derivation are provided in Appendix A.2.

$$bT_0 < ln(1+bT_0) \Leftrightarrow Var_{sys}(\bar{y}) > Var_{poi}(\bar{y}) \tag{4.53}$$

$$bT_0 > ln(1+bT_0) \Leftrightarrow Var_{sys}(\bar{y}) < Var_{poi}(\bar{y}) \tag{4.54}$$

In this thesis, the sampling period T_0 is specified as 1 second, then $bT_0 \approx \hat{b}T_0 = 0.086743$, $\ln(1+bT_0) \approx \ln(1+\hat{b}T_0) = 0.0832$, so we have $bT_0 > \ln(1+bT_0)$. Based on the theoretical analysis in Equation 4.54, we can obtain that $Var_{sys}(\bar{y}) < Var_{poi}(\bar{y})$, i.e., the timer-based systematic sampling should perform better than the timer-based Poisson sampling.

4.3.2.1 Simulation Results

Simulations for the two timer-based sampling methods are performed respectively. Each simulation is performed for 222 sampling rounds (i.e., each simulation is repeated 222 times). Then the mean packet delay and the variance of packet delay of the sample trace from the 222 systematic sampling rounds and the 222 Poisson sampling rounds are calculated. The main results are shown in Table 4.5. We can see that the timer-based systematic sampling indeed produces better performance than the timer-based Poisson sampling. Figure 4.7 shows the absolute error of the estimated mean and Figure 4.8 shows the absolute error of the estimated variance for the 222 sampling rounds, which demonstrates that the timer-based systematic sampling has both a lower absolute error of the estimated mean and a lower absolute error of the estimated variance. In conclusion, the timer-based systematic sampling has higher accuracy than the timer-based Poisson sampling, which agrees with the theoretical analysis.

Table 4.5
Main simulation results of timer-based systematic sampling and timer-based Poisson sampling (true values are: $\mu = 86.824$ ms, $\sigma^2 = 8529$).

Sampling method	M	$AMean = \frac{1}{M}\sum \hat{\mu}_i$	$AVar = \frac{1}{M}\sum s_i^2$	$MSE = \frac{1}{M}\sum(\hat{\mu}_i - \mu)^2$
Systematic	222	74.803 ms	5996	145
Poisson	222	64.998 ms	4710	478

4.4 Stratified Sampling *vs* Simple Random Sampling

In this section, theoretical comparisons of the variance of the sample mean are performed between simple random sampling, stratified sampling with proportional allocation and stratified sampling with optimum allocation. The variances of the sample mean for the three schemes are denoted by $Var_{ran}(\bar{y})$, $Var_{prop}(\bar{y})$ and $Var_{opt}(\bar{y})$ respectively.

For stratified sampling, it can be shown that the variance of the parent population

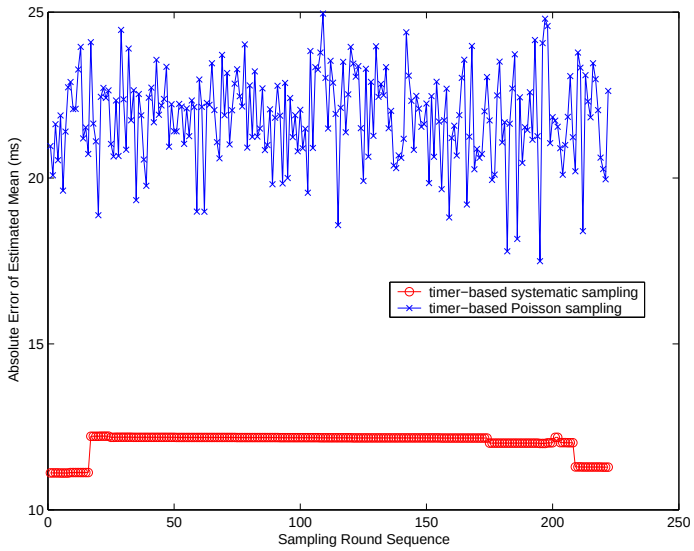


Figure 4.7 Comparison of Absolute Error of Estimated Mean between timer-based systematic sampling and timer-based Poisson sampling. Sampling interval: 1 second, parent traffic duration: 2600 seconds, sampling rounds: 222.

(i.e., the true variance) is related to the variances in each stratum by:

$$\sigma^2 = \frac{1}{N-1}\sum_{i=1}^{N}(y_i-\mu)^2 = \frac{1}{N-1}\sum_{l=1}^{L}\sum_{i=1}^{N_l}(y_{li}-\mu)^2, \tag{4.55}$$

$$= \frac{1}{N-1}\sum_{l=1}^{L}\sum_{i=1}^{N_l}[(y_{li}-\mu_l)+(\mu_l-\mu)]^2, \tag{4.56}$$

$$= \frac{1}{N-1}\sum_{l=1}^{L}(N_l-1)\sigma_l^2 + \frac{1}{N-1}\sum_{l=1}^{L}N_l(\mu_l-\mu)^2, \tag{4.57}$$

where σ_l^2 is defined in Equation 4.3. The details of the derivation are provided in Appendix A.3.

Multiplying both sides of Equation 4.57 by a common factor $\frac{1}{n}(1-\frac{n}{N})$, where $(1-\frac{n}{N})$ is the finite population correction (fpc) factor, it can be obtained that:

$$\frac{1}{n}(1-\frac{n}{N})\sigma^2 = \frac{1}{n}(1-\frac{n}{N})\frac{1}{N-1}\sum_{l=1}^{L}(N_l-1)\sigma_l^2 + \frac{1}{n}(1-\frac{n}{N})\frac{1}{N-1}\sum_{l=1}^{L}N_l(\mu_l-\mu)^2. \tag{4.58}$$

Applying the approximation in Equation 4.4, it can be obtained that:

$$\frac{1}{n}(1-\frac{n}{N})\sigma^2 = \frac{1}{n}(1-\frac{n}{N})\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l^2 + \frac{1}{nN}(1-\frac{n}{N})\sum_{l=1}^{L}N_l(\mu_l-\mu)^2. \tag{4.59}$$

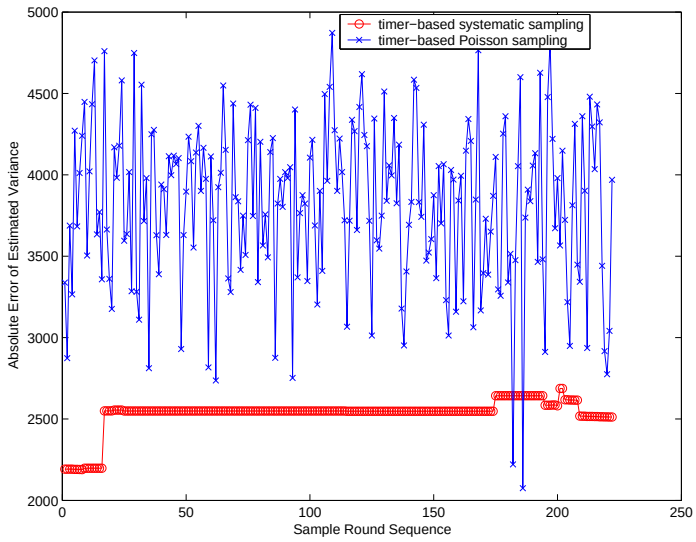


Figure 4.8 Comparison of Absolute Error of Estimated Variance between timer-based systematic sampling and timer-based Poisson sampling. Sampling interval: 1 second, parent traffic duration: 2600 seconds, sampling rounds: 222.

It should be noticed that the approximation in Equation 4.4 naturally leads to the approximation $N - 1 \approx N$.

4.4.1 Stratified Sampling with Proportional Allocation

For simple random sampling, it has been shown in Chapter 3 that the variance of the sample mean is

$$Var_{ran}(\bar{y}) = \frac{1}{n}(1 - \frac{n}{N})\sigma^2. \tag{4.60}$$

For stratified random sampling, the variance of the sample mean is given by [30, pp. 91]

$$Var_{st}(\bar{y}) = \sum_{l=1}^{L}(\frac{N_l}{N})^2(\frac{N_l - n_l}{N_l})\frac{\sigma_l^2}{n_l}. \tag{4.61}$$

If proportional allocation (i.e., the sample size in each stratum is proportional to the parent size in that stratum [30, pp. 91]) is used, then

$$n_l = n\frac{N_l}{N}. \tag{4.62}$$

The variance of the sample mean becomes:

$$Var_{prop}(\bar{y}) = \frac{1}{n}(1 - \frac{n}{N})\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l^2. \tag{4.63}$$

To compare the performance of different sampling techniques, I fix the total sample size and compare the variance of the sample mean under different sampling techniques. Comparing Equation 4.59 and 4.60 with Equation 4.63, it can be seen that the variance of the sample mean for simple random sampling is related to the variance of the sample mean for stratified sampling with proportional allocation by:

$$Var_{ran}(\bar{y}) = Var_{prop}(\bar{y}) + \frac{1}{n}(1 - \frac{n}{N})\frac{1}{N}\sum_{l=1}^{L}N_l(\mu_l - \mu)^2. \tag{4.64}$$

Applying the approximation $N_l - 1 \approx N_l$, the sampling gain of the stratified sampling with proportional allocation is given by:

$$\Delta_{prop} = var_{ran}(\bar{y}) - var_{prop}(\bar{y}) = \frac{1}{nN}(1 - \frac{n}{N})\sum_{l=1}^{L}N_l(\mu_l - \mu)^2 \geq 0. \tag{4.65}$$

The sampling gain Δ_{prop} is positive, which indicates that performance improvement (i.e., better sampling accuracy) can be achieved when moving from simple random sampling to stratified sampling with proportional allocation.

4.4.2 Stratified Sampling with Optimum Allocation

If stratified sampling with optimum allocation is used, the sample size in the l-th stratum is given by [30, pp. 97]:

$$n_l = \frac{nN_l\sigma_l}{\sum_{k=1}^{L}N_k\sigma_k}. \tag{4.66}$$

The variance of the sample mean can be obtained by Equation 4.61 and 4.66:

$$Var_{opt}(\bar{y}) = \frac{1}{nN^2}(\sum_{l=1}^{L}N_l\sigma_l)(\sum_{l=1}^{L}N_l\sigma_l) - \frac{1}{N}\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l^2, \tag{4.67}$$

$$= \frac{1}{n}(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)^2 - \frac{1}{N}\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l^2. \tag{4.68}$$

The sampling gain of the stratified sampling with optimum allocation in comparison with the stratified sampling with proportional allocation is given by Equation 4.69. The details of the derivation are provided in Appendix A.4:

$$Var_{prop}(\bar{y}) - Var_{opt}(\bar{y}) = \frac{1}{nN}\sum_{l=1}^{L} N_l(\sigma_l - \bar{\sigma}_l)^2 \geq 0, \tag{4.69}$$

where $\bar{\sigma}_l$ is:

$$\bar{\sigma}_l = \sum_{l=1}^{L} \frac{N_l}{N}\sigma_l. \tag{4.70}$$

As the sampling gain is always positive, stratified sampling with optimum allocation achieves better performance than stratified sampling with proportional allocation. Therefore with the same sample size n, ignoring the fpc, the sampling gain of the stratified sampling with optimum allocation in comparison with simple random sampling is:

$$\Delta_{opt} = Var_{ran}(\bar{y}) - Var_{opt}(\bar{y}) = \frac{1}{nN}[\sum_{l=1}^{L} N_l(\mu_l - \mu)^2 + \sum_{l=1}^{L} N_l(\sigma_l - \bar{\sigma}_l)^2] \geq 0. \tag{4.71}$$

Equation 4.71 demonstrates that stratified sampling with optimum allocation performs better than simple random sampling.

Finally, I conduct simulations for stratified sampling with optimum allocation for 222 sampling rounds. The parameters used to calculate the stratum sample size n_l are all true values from the parent delay trace, and the stratum size is specified as 50 seconds. Simulation results are shown in Table 4.6, Figure 4.9 and 4.10. The results indicate that the stratified sampling with optimum allocation achieves the best estimation accuracy, which agrees with our theoretical analysis.

4.5 Impact of Packet Size on Delay Measurements

In the earlier sections, I have investigated the performance of different sampling techniques without considering the effect of packet size. In order to find out the effects of packet size on the sampling accuracy, I deploy active sampling tests using Opnet simulation with two different sample traces. One sample trace has a constant packet

Table 4.6
Main simulation results of stratified sampling with optimum allocation (true values are: $\mu = 86.824$ ms, $\sigma^2 = 8529$)

Sampling method	M	$AMean = \frac{1}{M}\sum \hat{\mu}_i$	$AVar = \frac{1}{M}\sum s_i^2$	$MSE = \frac{1}{M}\sum(\hat{\mu}_i - \mu)^2$
Systematic	222	74.803 ms	5996	145
Poisson	222	64.998 ms	4710	478
Stratified with optimum allocation	222	88.023 ms	8959	5

size of 128 bytes and the other has approximately the same packet size distribution as the parent traffic trace. Figure 4.11 shows the cumulative distribution function (CDF) of the packet size from real traffic traces on different dates and times. We can see that the packet size distribution is approximately the same. We can also see that there are several typical values of packet size in the real traffic trace, e.g., 24 bytes, 550 bytes, 750 bytes and 1472 bytes. The probabilities of other packet sizes are comparatively small. Therefore, I use these typical values to build up the sample trace. Let Z denote a uniform distribution random variable, then the packet size that has the same CDF as the parent traffic trace can be obtained by:

- if $0 \leq z < 0.5$: packet size is 24 bytes;
- if $0.5 \leq z < 0.65$: packet size is 550 bytes;
- if $0.65 \leq z < 0.78$: packet size is 572 bytes;
- if $0.78 \leq z < 0.80$: packet size is 750 bytes;
- if $0.80 \leq z \leq 1.0$: packet size is 1472 bytes.

Where the values 0.5, 0.65, 0.78 and 0.80 are obtained from empirical observation using Figure 4.11. Figure 4.12 shows the CDFs of the constructed sample packet size and the parent packet size. They are approximately the same.

Then I perform active sampling tests using this constructed sample trace and the constant packet size sample trace respectively. The sampling goal is to estimate the mean packet delay and the variance of packet delay. Table 4.7 shows the results of

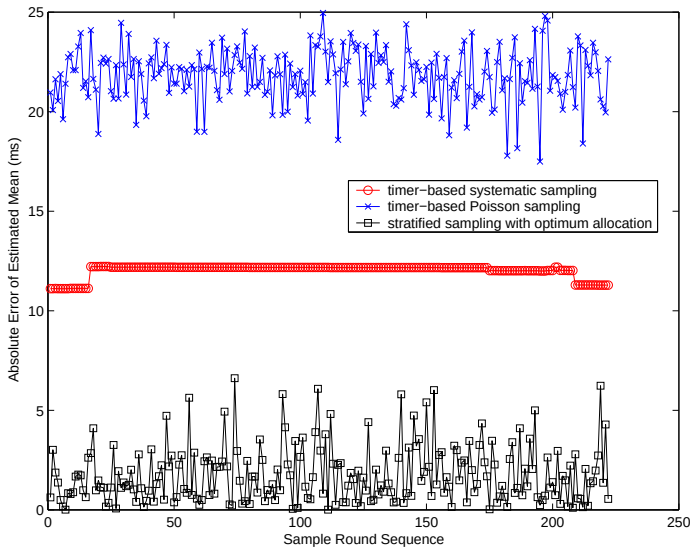


Figure 4.9 Comparison of Absolute Error of Estimated Mean between timer-based systematic sampling, timer-based Poisson sampling and stratified sampling with optimum allocation. Sampling interval: 1 second, parent traffic duration: 2600 seconds, stratum size: 50 seconds, sampling rounds: 222.

the two sampling tests. The sampling results indicate that the estimation of sample

Table 4.7
Main simulation results of active sampling by Opnet simulation. Sample A is the sample with constant packet size distribution and sample B is the sample with approximately the same packet size distribution as the parent trace.

Property	Min. Delay	Max. Delay	Mean Delay	Median Delay	Var. of Delay	StdDev of Delay
Parent	41.092 ms	1413.05 ms	86.024 ms	45.2 ms	8529	92.354
Sample A	50.7 ms	51.6 ms	50.9 ms	50.9 ms	0.0085	0.092
Sample B	40.2 ms	310.4 ms	94.7 ms	40.7 ms	11600	107.8

A is significantly biased, and sample B gives high accuracy for estimating the mean packet delay and the variance of packet delay. Since the packet size distribution of the real traffic trace are approximately the same, we can use the constructed sample trace to estimate the mean packet delay and the variance of packet delay, which can improve the accuracy of estimation.

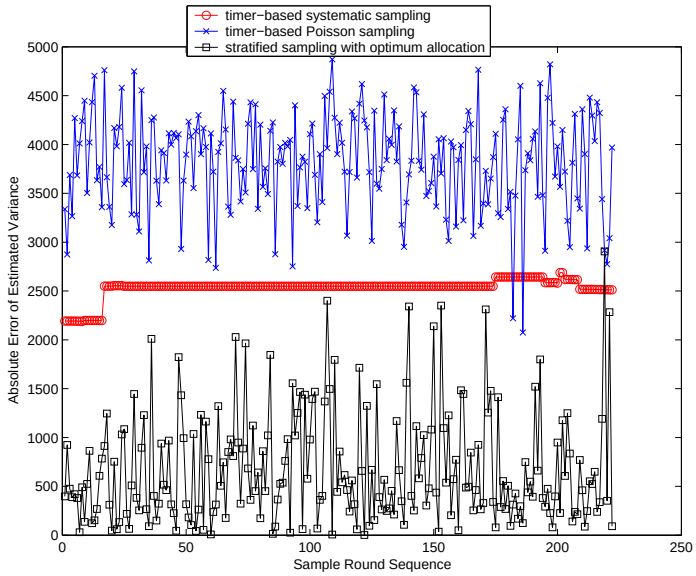


Figure 4.10 Comparison of Absolute Error of Estimated Variance between timer-based systematic sampling, timer-based Poisson sampling and stratified sampling with optimum allocation. Sampling interval: 1 second, parent traffic duration: 2600 seconds, stratum size: 50 seconds, sampling rounds: 222.

4.6 Summary

In this chapter, I compared the performance of different sampling techniques. Performance comparison was conducted by comparing the variance of the sample mean of different sampling techniques under the constraint that the total sample sizes were the same. In addition, several other metrics were employed for the performance comparison, such as the absolute error of the estimated mean and the absolute error of the estimated variance.

I first compared the performance of the count-based systematic sampling and the count-based simple random sampling. Based on the parent delay trace, the count-based systematic sampling achieved approximately the same performance as the count-based simple random sampling, with a marginal improvement. Secondly, I compared the performance of timer-based systematic sampling and timer-based Poisson sampling. The theoretical analysis showed that the difference between their performance was related to the autocorrelation of packet delay ρ of the parent delay trace

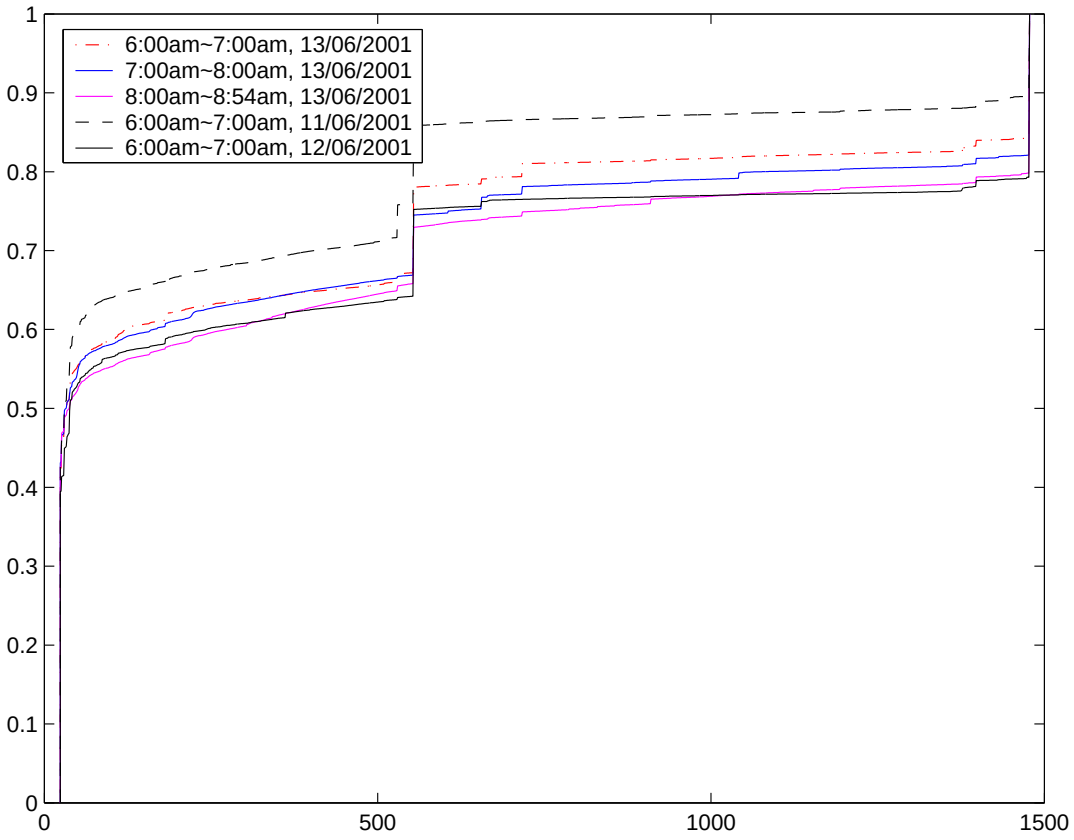


Figure 4.11 CDF of packet size of different time and date

and the sampling interval T_0. From the autocorrelation of delay ρ and the sampling interval in our study (i.e., 1 second), the timer-based systematic sampling should achieve better performance than the timer-based Poisson sampling. Simulation results demonstrated that the timer-based sampling indeed produced a better accuracy of estimate than the timer-based Poisson sampling. Therefore, the simulation results complied with the theoretical analysis. Thirdly, I compared the performance between simple random sampling and stratified random sampling with proportional allocation and stratified random sampling with optimum allocation. The conclusion was obtained that stratified sampling with proportional allocation achieved better performance than simple random sampling, and stratified sampling with optimum allocation achieved the best performance. However, the better performance of the stratified sampling was achieved at the expense of the greater complexity of the system. As pointed out in Chapter 3.2.3, this higher complexity might compromise the

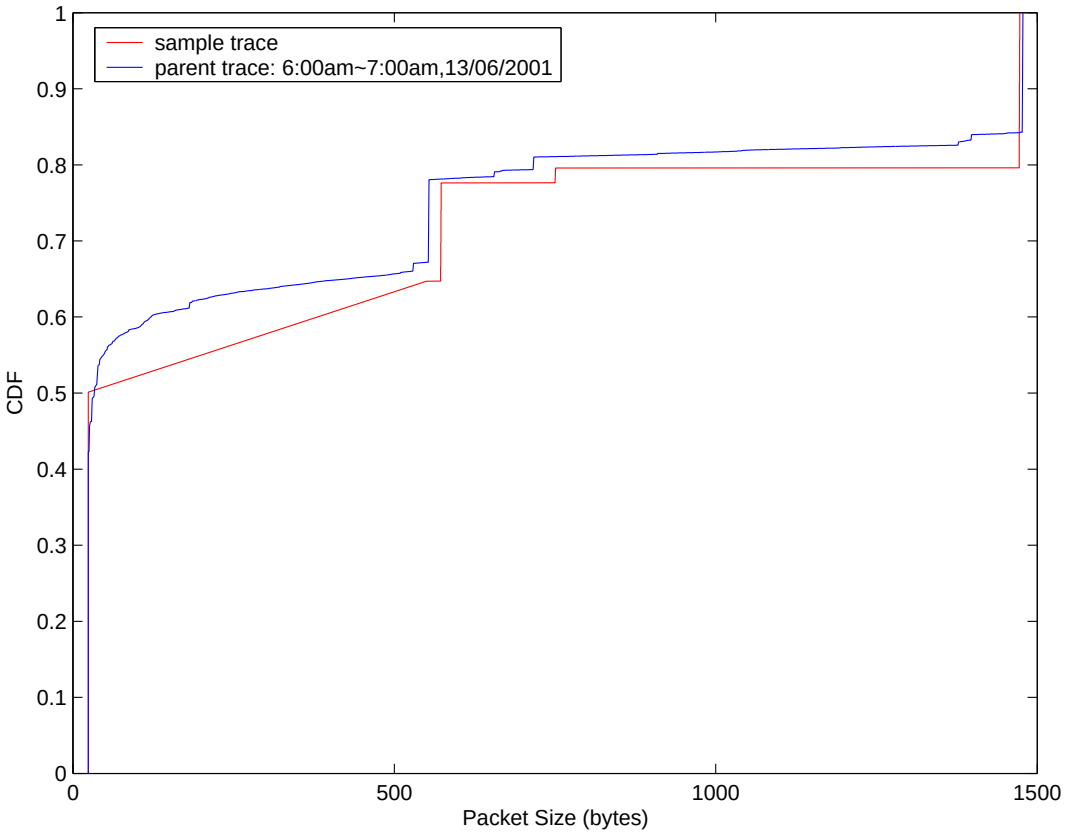


Figure 4.12 CDF of sample packet size and parent packet size.

advantage of the stratified sampling technique.

Chapter 5

Adaptive Stratified Sampling

5.1 Introduction

It has been shown in Chapter 4.4.2 that the stratified sampling with optimum allocation produces higher accuracy for estimating the mean value of packet delay and the variance of packet delay. However, the stratified sampling with optimum allocation requires extra information (e.g., standard deviation of packet delay within a stratum) from the parent delay trace in order to determine the sample size within each stratum. This has been shown in Equation 4.66. In real applications, these statistics are not known *a priori*. To address the challenge, I propose a novel adaptive stratified sampling scheme which can closely approach the performance of stratified sampling with optimum allocation.

As discussed in Chapter 3.2.4, adaptive sampling methods select samples adaptively according to values of the variable of interest observed during the sampling process. The key element of adaptive sampling is the prediction of future behaviour based on the observed behaviour [41]. Hernandez *et al.* employe a linear prediction (LP) algorithm in their adaptive sampling method to measure the network throughput [31]. Choi *et al.* use an autoregressive model for network load change detection [42]. Ma *et al.* employe a sample size estimation algorithm in their sampling scheme for network performance measurement [41]. In this chapter, I propose an adaptive stratified sampling scheme, which employs a least-mean-square (LMS) linear prediction algo-

rithm to predict the standard deviation of packet delay from past observations. Then the sample size for the next stratum is calculated from the predicted value of the standard deviation.

Firstly, a brief introduction to the LMS algorithm is presented. Secondly, a detailed description of the adaptive stratified sampling algorithm is presented. Finally, a performance comparison between the proposed sampling strategy and two timer-based sampling methods (e.g., timer-based systematic sampling) is made.

5.2 Least-mean-square Algorithm

The LMS algorithm is one of the most widely used adaptive linear algorithms. A significant feature of the LMS algorithm is its simplicity. It does not require measurements of the correlation function, nor does it require matrix inversion. The adaptive mechanism enables it to approximate the steepest descent algorithm automatically from sample to sample. Figure 5.1 shows the architecture of the LMS algorithm.

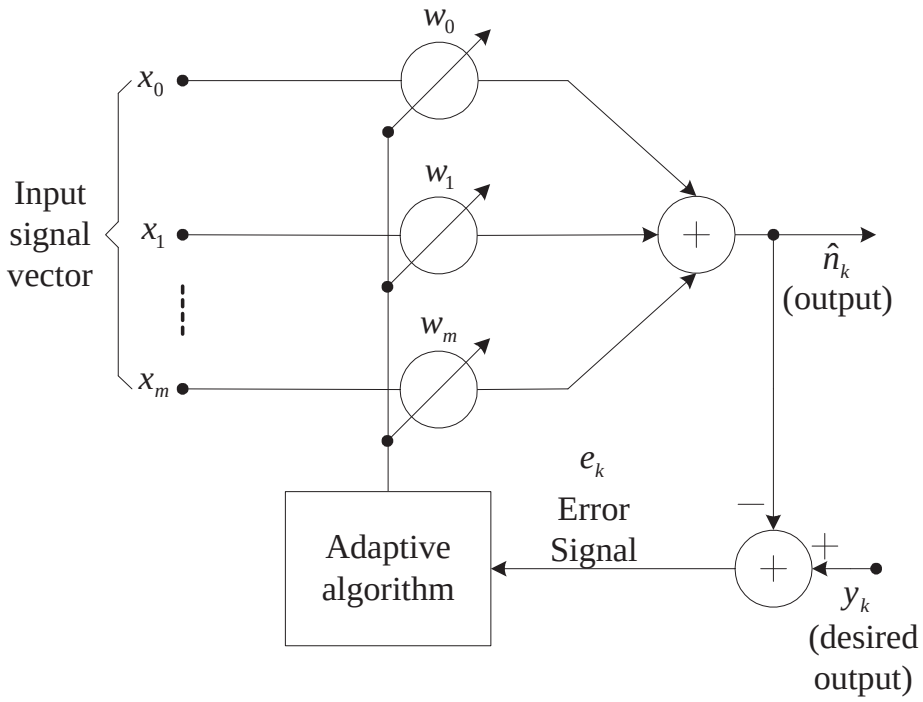


Figure 5.1 Architecture of LMS algorithm

The computational procedure for the LMS algorithm is listed below: [43, pp. 655]:

- Compute filter output

$$\hat{y}_k = \sum_{i=0}^{m-1} w_k(i) y_{k-1-i} = \mathbf{W}_k^T \mathbf{Y}(k). \tag{5.1}$$

  where m is the order of the predictor, $\mathbf{Y}(k)$ is the input vector and $\mathbf{W}_k$ is the prediction coefficient vector.

$$\mathbf{Y}(k) = [y_{k-1}, y_{k-2}, ..., y_{k-m}]^T, \tag{5.2}$$

$$\mathbf{W}_k = [w_k(0), w_k(1), ..., w_k(m-1)]^T. \tag{5.3}$$

  Initially, each weight $w_k(i)$ is set to an arbitrary fixed value.

- Compute the prediction error

$$e_k = y_k - \hat{y}_k. \tag{5.4}$$

- Update the coefficient vector

$$\mathbf{W}_{k+1} = \mathbf{W}_k + 2\upsilon e_k \mathbf{Y}(k), \tag{5.5}$$

  where υ is the step size.

The LMS algorithm does not require knowledge of **P** and **R**, but instead uses their instantaneous estimates, where $\mathbf{P} = E[y_k \mathbf{Y}(k)]$ is the cross-correlation vector between the input vector $\mathbf{Y}(k)$ and the desired output y_k, and $\mathbf{R} = E[\mathbf{Y}(k)\mathbf{Y}^T(k)]$ is the $m \times m$ correlation matrix. The weight vector $\mathbf{W}_k$ is only an estimate, but it is updated and improved from sample to sample. Eventually, the weights converge; the condition for convergence is:

$$0 < \upsilon < \frac{1}{\lambda_{max}}, \tag{5.6}$$

where λ_{max} is the maximum eigenvalue of the input data covariance matrix **R**. In practice, $\mathbf{W}_k$ never reaches the theoretical optimums, but fluctuates about it. A conservative estimation for λ_{max} is the input vector energy $\|\mathbf{Y}(k)\|^2$ [44, pp. 303].

$$0 < \upsilon < \frac{1}{\|\mathbf{Y}(k)\|^2}. \tag{5.7}$$

The MSE of prediction will decrease to its minimum value (denoted as J_{min}) if the weight vector $\mathbf{W}_k$ reaches the optimum weights $\mathbf{W}_{opt} = \mathbf{R}^{-1}\mathbf{P}$. However, in real applications, $\mathbf{W}_k$ is only an estimate and cannot reach $\mathbf{W}_{opt}$. So the MSE of prediction cannot exactly reach J_{min}. The actual MSE of prediction by LMS algorithm consists of three components [44, pp. 267]:

$$J(k) = E[|e_k|^2] = J_{min} + J_{tr}(k) + J_{ex}(\infty), \tag{5.8}$$

where $J_{tr}(k)$ is transient component of the MSE and $J_{ex}(\infty)$ is steady-state excess MSE.

The transient component $J_{tr}(k)$ dies out if and only if the step size υ satisfies Equation 5.7.

The excess MSE $J_{ex}(k)$ can be approximated as:

$$J_{ex}(k) \approx \upsilon J_{min} \sum_{i=1}^{m} \lambda_i. \tag{5.9}$$

where λ_i are the eigenvalues of $\mathbf{R}$.

Hence, given step size υ satisfying Equation 5.7, the MSE of LMS algorithm can be simplified as:

$$J(k) \approx J_{min} + \upsilon J_{min} \sum_{i=1}^{m} \lambda_i = J_{min}(1 + \upsilon \sum_{i=1}^{m} \lambda_i). \tag{5.10}$$

5.3 Adaptive Stratified Sampling Algorithm

5.3.1 Prediction of Sample Size within Strata

Obtaining the sample size n_l within each stratum is the key phase in stratified sampling with optimum allocation. n_l is calculated by Equation 4.66 as:

$$n_l = \frac{nN_l\sigma_l}{\sum_{k=1}^{L} N_k\sigma_k}. \tag{5.11}$$

In order to estimate n_l, an assumption is made that the parent population size N_l is

approximately the same in each stratum, i.e.,

$$\frac{N_l}{N_k} \approx 1, \quad l \neq k. \tag{5.12}$$

This assumption is valid when the parent population size N_l is very large and the time duration of the stratum is a constant. This assumption has been validated using the real traffic trace. Figure 5.2 shows the ratio N_k/N_1 of the real traffic trace with stratum size $= 50, 100, 130$ and 200 seconds respectively, where $N_k,\ k = 1, 2, ..., L$ is the total number of packets within the k-th stratum of the real traffic trace and N_1 is the total number of packets within the 1-st stratum of the real traffic trace. We can see that the ratio N_k/N_1 is bounded in the interval $[0.8, 1.2]$.

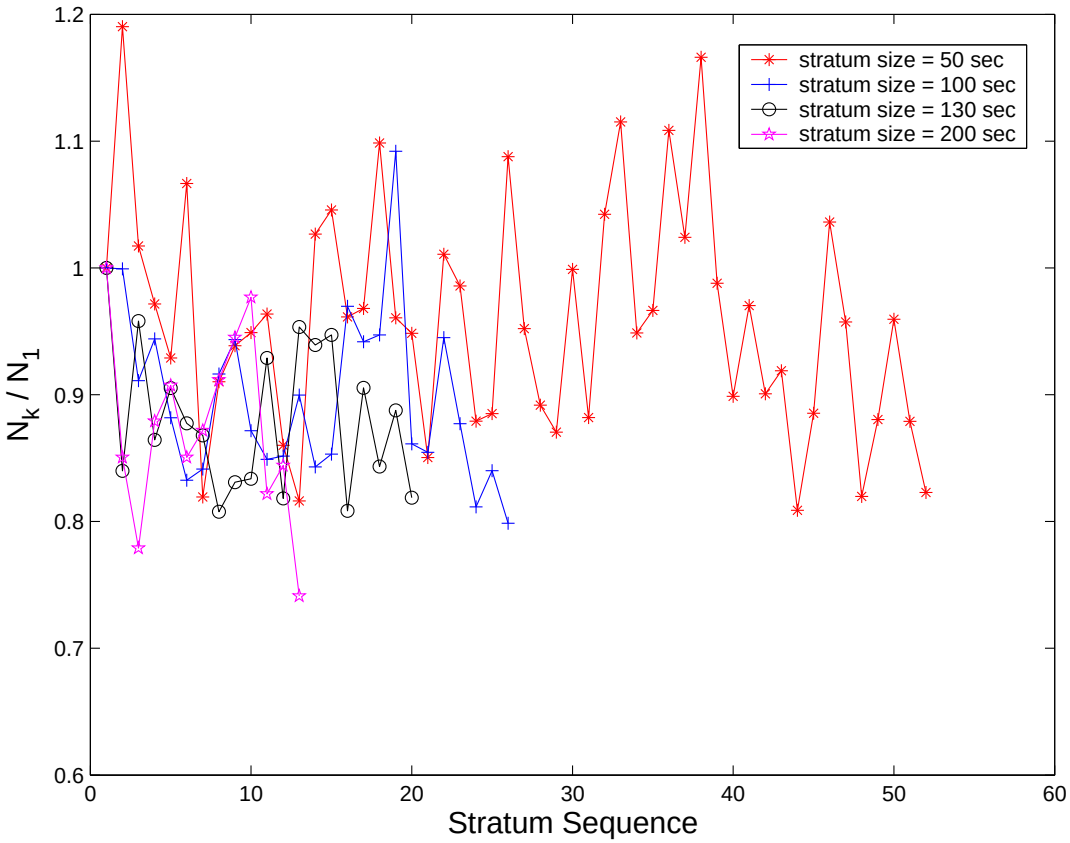


Figure 5.2 Ratio of packet number between different strata

Then Equation 5.11 can be simplified as:

$$n_l \approx \frac{n\sigma_l}{\sum_{k=1}^{L}\sigma_k} = \frac{n\sigma_l}{L\bar{\sigma}_s} = \varphi\sigma_l, \tag{5.13}$$

$$\varphi = \frac{n}{\sum_{k=1}^{L}\sigma_k} = \frac{n}{L\bar{\sigma}_s}, \tag{5.14}$$

where $\bar{\sigma}_s = \frac{1}{L}\sum_{k=1}^{L}\sigma_k$ is the mean value of the standard deviation of delay among all the strata. In real applications, φ is a proportionality constant which determines the sampling rate. φ can be chosen empirically and a larger φ will produce s a higher sampling rate.

The next step is to find a way to predict σ_l. The LMS algorithm is employed to predict σ_l from the past observations (e.g., σ_{l-1},σ_{l-2}). From the LMS algorithm, the l-th standard deviation σ_l is given by:

$$\hat{\sigma}_l = \sum_{i=0}^{m-1} w_l(i)\sigma_{l-1-i}. \tag{5.15}$$

The prediction error e_l is:

$$e_l = \sigma_l - \hat{\sigma}_l, \tag{5.16}$$

and the weights are updated by:

$$w_{l+1}(i) = w_l(i) + 2ve_l\sigma_{l-1-i} \quad i = 0, 1, ..., m-1. \tag{5.17}$$

Since the standard deviation of delay σ_l is the true value of the parent delay trace, which we cannot obtain in practice, it must be replaced by its estimated value s_l, where s_l is the standard deviation of packet delay of the sample delay trace. Although s_l^2 (variance of packet delay of sample delay trace) is an unbiased estimate of σ_l^2, the estimate of the standard deviation of parent trace σ_l by the standard deviation of sample trace s_l is slightly biased. Fortunately, in most applications the bias is marginal [30, pp. 25]. For the proposed adaptive sampling scheme, I use the standard deviation of packet delay s_l of sample trace to estimate the standard deviation of packet delay σ_k of parent trace.

Hence, the estimator $\hat{n}_l$ of sample size for the l-th stratum can be computed by:

$$\hat{n}_l \quad = \quad \varphi\hat{s}_l, \tag{5.18}$$

$$\hat{s}_l = \sum_{i=0}^{m-1} w_l(i) s_{l-1-i}, \tag{5.19}$$

$$e_l = s_l - \hat{s}_l, \tag{5.20}$$

$$w_{l+1}(i) = w_l(i) + 2ve_l s_{l-1-i} \quad i = 0, 1, ..., m-1. \tag{5.21}$$

The next step is to determine the predictor order m and the initial value of weight vector. m can be determined based on the autocorrelation of the standard deviation of packet delay. The best performance is obtained when the initial weights are specified as the stable values of weights when the weights approximately converges. To establish the validity of prediction, we separate the sampling procedure into two parts. In the first part, we use the required statistics (i.e., autocorrelation of the standard deviation of packet delay) obtained from sampling to determine the predictor order m, and train the weights at the same time of sampling. The main objective in the first part is to train the weight values. Then in the second part, we use the chosen predictor order m and the stable weights obtained in the first part to perform sampling. In this thesis, the parent delay trace is not long enough to be separated into two parts. To address this problem, I choose another real traffic trace provided by the WAND research group to determine the predictor order m and the initial weights. This trace ("20010612-060000-e1.gz") was captured on June 12th, 2001, one day before the day when the trace discussed in Chapter 4.2 was captured. A new delay trace is generated by importing the first 2600-second part of this trace into Opnet Modeler. Figure 5.3 shows the autocorrelation of the standard deviation of packet delay of the new delay trace. For the stratum size of 100 seconds, the autocorrelation is significant at lags of $1, 3, 4$ and 5, so an order of 5 must be chosen. For the stratum size of 50 seconds, the autocorrelation is significant at lags of $1, 2, 3$ and 4, and it declines to approximately zero at a lag of 5. Therefore, an order of 4 is an appropriate choice. There is a simple way to diagnose whether the order m selected is appropriate. After setting the order m to a certain value (e.g., $m = 4$ for stratum size of 50 seconds), we check the autocorrelation of the residual error e_l. If the autocorrelation of error e_l is significant, then the order is inappropriate, and it must be modified and improved.

Take a stratum size of 50 seconds as an example, the new delay trace is separated into $2600/50 = 52$ strata, then we can obtain 52 standard deviations of packet delay.

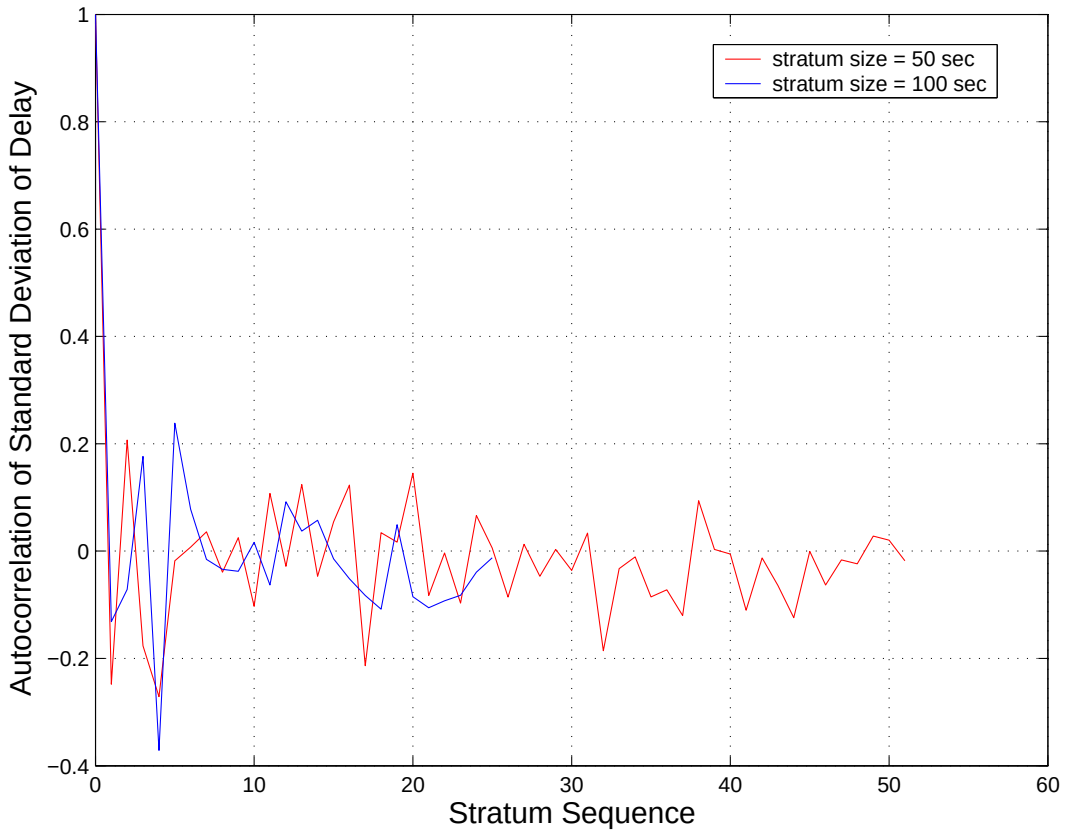


Figure 5.3 Autocorrelation of the standard deviation of delay

I first set the weights to $w_l(0) = w_l(1) = w_l(2) = w_l(3) = 0.25$, then perform the prediction on the 52 standard deviations of packet delay. The 52 values of standard deviation is used repeatedly until the weight vector approximately converges. After each repetition, the initial weight vector for the next repetition is replaced by the weight vector at the end of the current cycle. Figure 5.4 shows the variations and convergence of the weights. The presence of the periodic peak is due to the prediction being repeatedly performed on the same data series (i.e., the 52 standard deviations of packet delay). When the weight vector approximately converges, the stable weight vector is selected as the initial weight vector for prediction in the future. As shown in Figure 5.4, the stable weights are: $w_l(0) = 0.256643$, $w_l(1) = 0.210456$, $w_l(2) = 0.209488$ and $w_l(3) = 0.260129$. The step size υ is 0.02. Then I use this initial weights to perform prediction on the delay trace discussed in Chapter 4.2. Figure 5.5 shows the variations of the weights. The autocorrelation of prediction error e_l

is also computed and shown in Figure 5.6. We can see that the prediction error e_l is approximately independent, which indicates a good performance of the prediction algorithm.

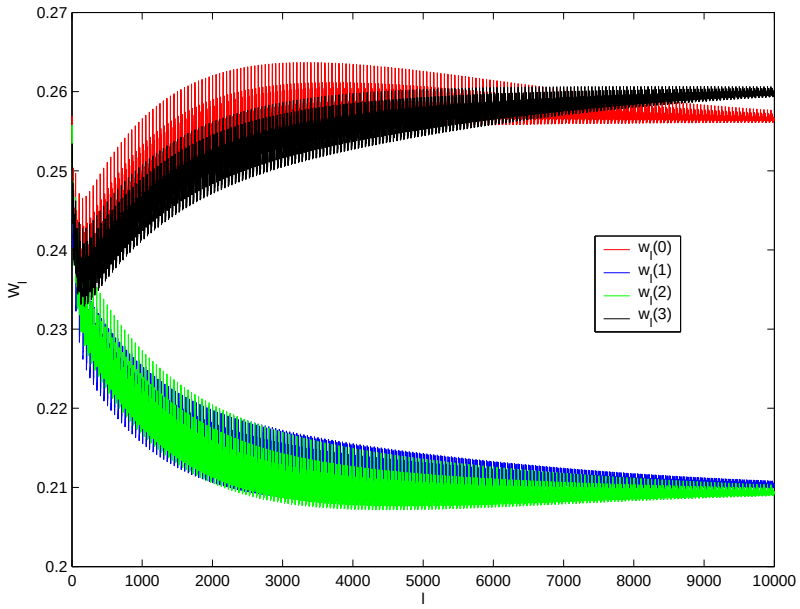


Figure 5.4 Variations and convergence of the weights. Initial value: $w_l(0) = w_l(1) = w_l(2) = w_l(3) = 0.25$.

5.3.2 Prediction Error

From Equation 5.18, 5.19 and 5.20, we can conclude that there are mainly two parts that contribute to the prediction errors when estimating n_l by $\hat{n}_l$; they are:

- estimation error in using s_l as an approximation of σ_l;
- prediction error in using LMS algorithm to predict s_l.

For the first item: we know that for a sample with ω elements the statistic $\eta = \frac{(\omega-1)s^2}{\sigma^2}$ has a χ^2 distribution with $(\omega - 1)$ degrees of freedom [40, pp. 216]. Let λ_1 and λ_2 denote the two critical values of the χ^2 distribution, which are determined by the confidence level $1 - \alpha$, then the confidence interval of the variance of the parent

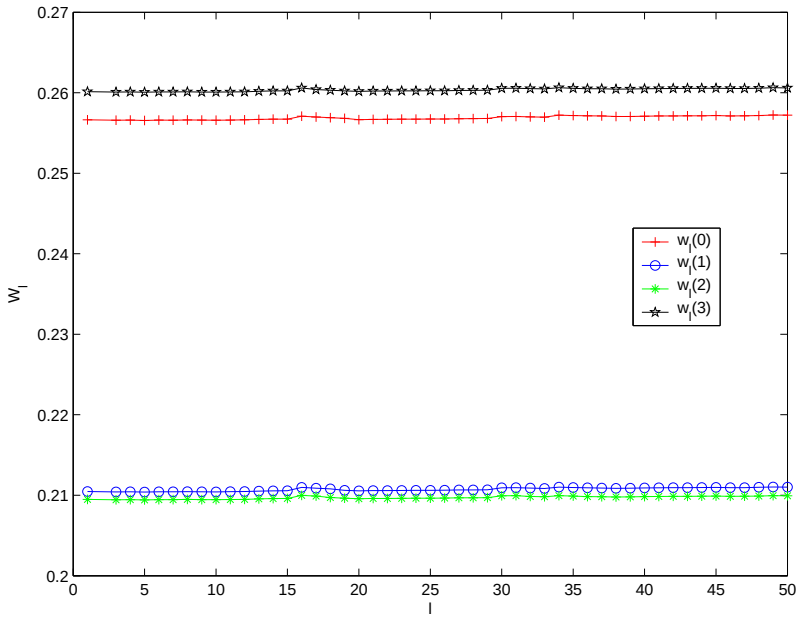


Figure 5.5 Variations of the weights. Initial value: $w_l(0) = 0.256643$, $w_l(1) = 0.210456$, $w_l(2) = 0.209488$ and $w_l(3) = 0.260129$.

population σ^2 is [40, pp. 216]:

$$\frac{(\omega-1)s^2}{\lambda_2} \leq \sigma^2 \leq \frac{(\omega-1)s^2}{\lambda_1}, \tag{5.22}$$

where s^2 is the variance of the sample.

From Equation 5.22, we can obtain the confidence interval of the standard deviation of the parent population σ,

$$s\sqrt{\frac{(\omega-1)}{\lambda_2}} \leq \sigma \leq s\sqrt{\frac{(\omega-1)}{\lambda_1}}, \tag{5.23}$$

where s is the standard deviation of the sample.

For the second item: the prediction error is e_l in Equation 5.20. Let e_{max} denote the maximum absolute value of e_l, then:

$$s_l - e_{max} \leq \hat{s}_l \leq s_l + e_{max}. \tag{5.24}$$

In practice, e_{max} can be estimated as:

$$e_{max} \approx z_{\frac{\alpha}{2}}\sqrt{\frac{E[|e_l|^2]}{L}} = z_{\frac{\alpha}{2}}\sqrt{\frac{J(l)}{L}}, \tag{5.25}$$

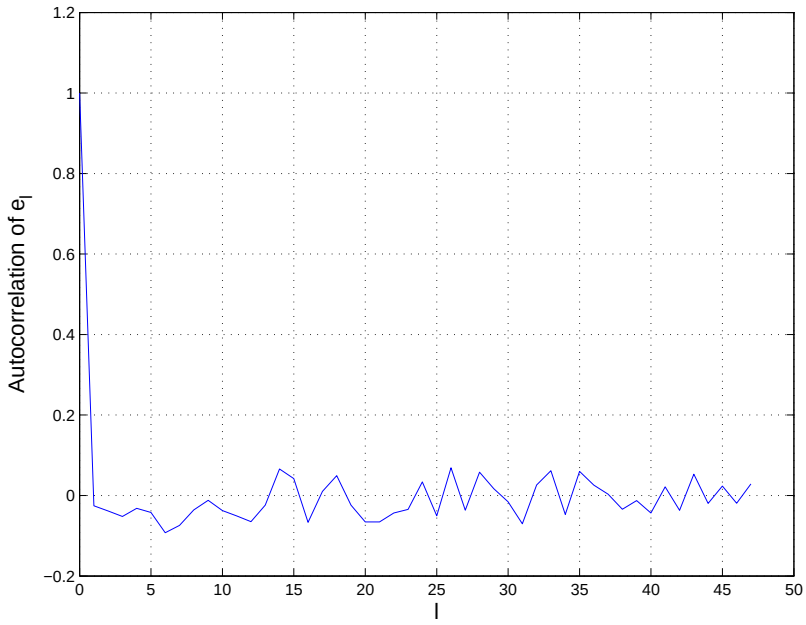


Figure 5.6 Autocorrelation of prediction error e_l

where $z_{\frac{\alpha}{2}}$ is the upper $\frac{\alpha}{2}$ quantile of the Gaussian distribution, L is the number of strata and $J(l)$ is defined in Section 5.2.

For the l-th stratum, the sample size within the stratum is n_l, which is estimated by $\hat{n}_l$. From Equation 5.23, we have:

$$\sqrt{\frac{\lambda_1}{\hat{n}_l - 1}} \leq \frac{s_l}{\sigma_l} \leq \sqrt{\frac{\lambda_2}{\hat{n}_l - 1}}. \tag{5.26}$$

The ratio between the sample size n_l and its estimate $\hat{n}_l$ is simplified by:

$$\frac{\hat{n}_l}{n_l} = \frac{\varphi \hat{s}_l}{\varphi \sigma_l} = \frac{\hat{s}_l}{\sigma_l}. \tag{5.27}$$

Hence, from Equation 5.24, 5.26 and 5.27, we can obtain the confidence bounds of the ratio $\frac{\hat{n}_l}{n_l}$ with a given confidence level $1 - \alpha$:

$$\sqrt{\frac{\lambda_1}{\hat{n}_l - 1}} - \frac{e_{max}}{\hat{s}_l} \leq \frac{\hat{n}_l}{n_l} \leq \sqrt{\frac{\lambda_2}{\hat{n}_l - 1}} + \frac{e_{max}}{\hat{s}_l}, \tag{5.28}$$

where λ_1 and λ_2 is determined by $1 - \alpha$.

The prediction errors may increase the expected variance of the sample mean, i.e., decrease the expected measurement accuracy of the adaptive sampling method. From

Equation 4.61, the actual variance of the sample mean using the predicted stratum sample size $\hat{n}_l$ is:

$$Var_{act}(\bar{y}) = \sum_{l=1}^{L}(\frac{N_l}{N})^2\frac{\sigma_l^2}{\hat{n}_l} - \sum_{l=1}^{L}(\frac{N_l}{N})^2\frac{\sigma_l^2}{N_l}. \tag{5.29}$$

The variance of the sample mean of stratified sampling with optimum allocation is given by Equation 4.68 in Chapter 4.4.2:

$$Var_{opt}(\bar{y}) = \frac{1}{n}(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)^2 - \frac{1}{N}\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l^2. \tag{5.30}$$

Then we can derive the relative error between $Var_{act}(\bar{y})$ and $Var_{opt}(\bar{y})$. The details of the derivation are provided in Appendix A.5.

$$\frac{Var_{act}(\bar{y}) - Var_{opt}(\bar{y})}{Var_{opt}(\bar{y})} = \frac{1}{n}\sum_{l=1}^{L}\frac{(\hat{n}_l - n_l)^2}{\hat{n}_l} = \frac{1}{n}\sum_{l=1}^{L}n_l\frac{(\phi_l - 1)^2}{\phi_l}, \tag{5.31}$$

where $\phi_l = \hat{n}_l/n_l$, which is bounded by Equation 5.28.

Therefore, the relative error of $Var_{act}(\bar{y})$ is related to the prediction error of the sample size. Let Λ denote the function $\Lambda(\phi) = \frac{(\phi-1)^2}{\phi} = \phi + 1/\phi - 2$, Figure 5.7 shows the graphics of $\Lambda(\phi)$. We can see that Λ can reach its maximum value at the lower bound or the upper bound of ϕ. Let ϕ_{max} denote the value at which Λ reaches its maximum value.

Then Equation 5.31 can be simplified as:

$$\begin{aligned}
\frac{Var_{act}(\bar{y}) - Var_{opt}(\bar{y})}{Var_{opt}(\bar{y})} &= \frac{1}{n}\sum_{l=1}^{L}n_l\frac{(\phi_l - 1)^2}{\phi_l}, && (5.32)\\
&\leq \frac{1}{n}\sum_{l=1}^{L}n_l\frac{(\phi_{max} - 1)^2}{\phi_{max}}, && (5.33)\\
&= \frac{(\phi_{max} - 1)^2}{\phi_{max}}\frac{1}{n}\sum_{l=1}^{L}n_l, && (5.34)\\
&= \frac{(\phi_{max} - 1)^2}{\phi_{max}}. && (5.35)
\end{aligned}$$

If $\phi_{max} = 0.9$, then the relative error between $Var_{act}(\bar{y})$ and $Var_{opt}(\bar{y})$ is bounded up to: 0.0111; if $\phi_{max} = 1.2$, then the relative error between $Var_{act}(\bar{y})$ and $Var_{opt}(\bar{y})$ is bounded up to: 0.0333.

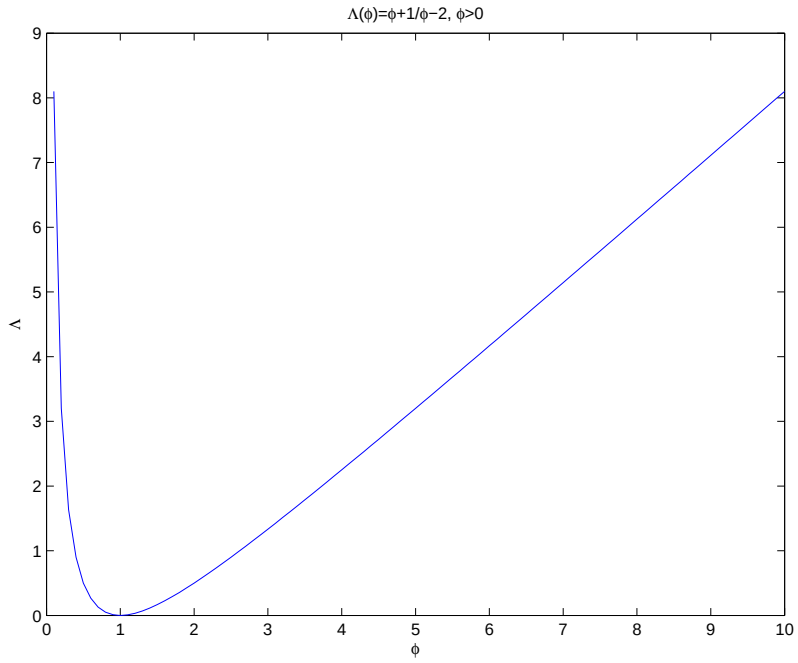


Figure 5.7 $\Lambda(\phi) = \phi + 1/\phi - 2$

5.3.3 Stratification Boundaries

The construction of strata is another key phase in stratified sampling methods. The aim in finding appropriate stratification boundaries is to make the variances of the interested variables (e.g., packet delay) within strata as small as possible.

The simplest way to decide the stratification boundaries is to divide the scale of the stratification variable into equal-sized intervals [45]. Usually, the stronger the correlation between the stratification variable and the survey variable is, the higher the measurement accuracy can be [29]. In [28], Zseby divides the parent population into different strata according to packet size. In the proposed adaptive sampling scheme, the sampling process is divided into fixed time intervals according to the autocorrelation of the survey variable (e.g., packet delay). The stronger the correlation between packet delays in an interval is, the more accurate the estimated mean delay will be.

Under a given sampling rate, the stratum size T_{strata} (i.e., time interval) should have an lower bound. Because if the stratum size is too small, the sample size within a

stratum may be not enough for estimation. A commonly used rule says that for a non-normal distributed parent population the sample size should be at least 50 [29]. Therefore, the mean sample size within stratum $\bar{n}_l$ needs to be greater than 50. In this thesis, the sampling rate is $1\ packet/second$, hence, the minimum stratum size is $50 \div 1 = 50$ seconds, i.e.,

$$T_{strata} \geq 50. \tag{5.36}$$

The number of strata L needs to satisfy the following equation:

$$L = \frac{T_{whole}}{T_{strata}} \leq \frac{T_{whole}}{50}, \tag{5.37}$$

where T_{whole} is the whole sampling duration.

In this thesis, the sampling duration is $T_{whole} = 2600$ seconds. Then the maximum number of strata is 52.

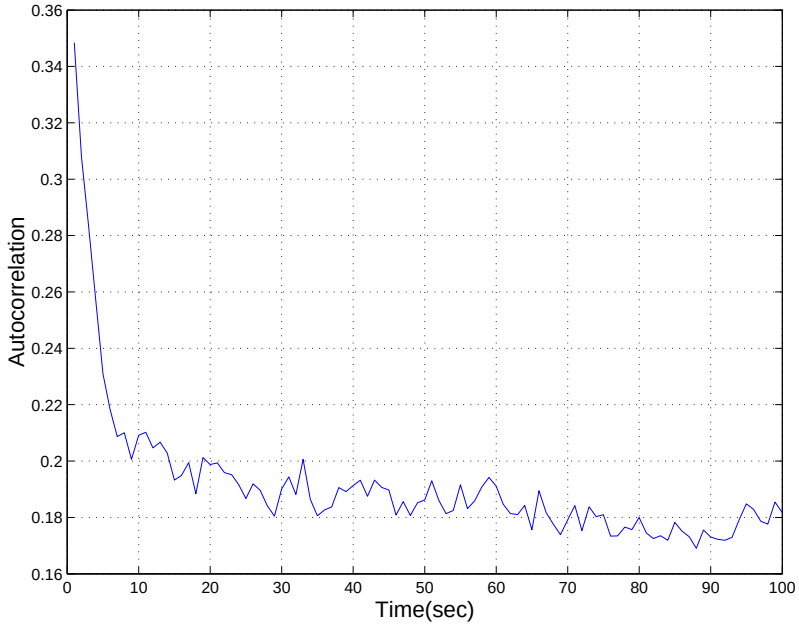


Figure 5.8 Autocorrelation of packet delay without the point $\rho_0 = 1$.

I determine the stratum size based on the autocorrelation of the packet delay such that packet delays classified into the same stratum have strong correlation. Figure 5.8 shows the autocorrelation of the packet delay of the parent delay trace. We can see that the delay correlation decreases significantly to the value 0.2 and it decreases

slowly after a lag of 20 seconds. A stratum size of 20 seconds is therefore a good choice for stratification. However, the previous discussion proves that the stratum size should be no less than 50 seconds. To determine an appropriate stratum size, I select three typical stratum sizes, i.e., 20 seconds, 50 seconds (minimum stratum size) and 90 seconds, and perform simulations respectively. Each simulation is performed for 222 sampling rounds. The simulation results are shown in Table 5.1, Figure 5.9 and 5.10. The results indicate that the stratum size of 50 seconds produces the best performance. Therefore, I select 50 seconds as the stratum size.

Table 5.1
Main simulation results of adaptive stratified sampling with different stratum size (true values are: $\mu = 86.824$ ms, $\sigma^2 = 8529$)

Stratum size	M	$AMean = \frac{1}{M}\sum \hat{\mu}_i$	$AVar = \frac{1}{M}\sum s_i^2$	$MSE = \frac{1}{M}\sum(\hat{\mu}_i - \mu)^2$
20 sec	222	83.834 ms	7808	19
50 sec	222	84.895 ms	8081	10
90 sec	222	84.470 ms	7953	18

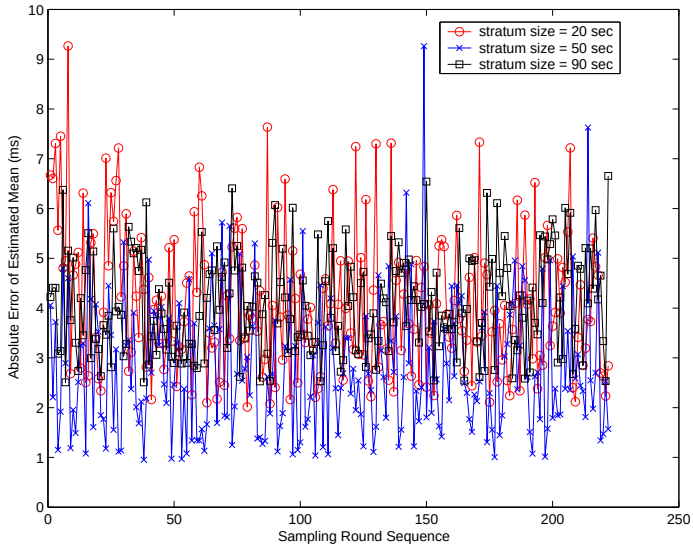


Figure 5.9 Absolute Error of Estimated Mean of adaptive stratified sampling with stratum size = 20, 50 and 90 seconds.

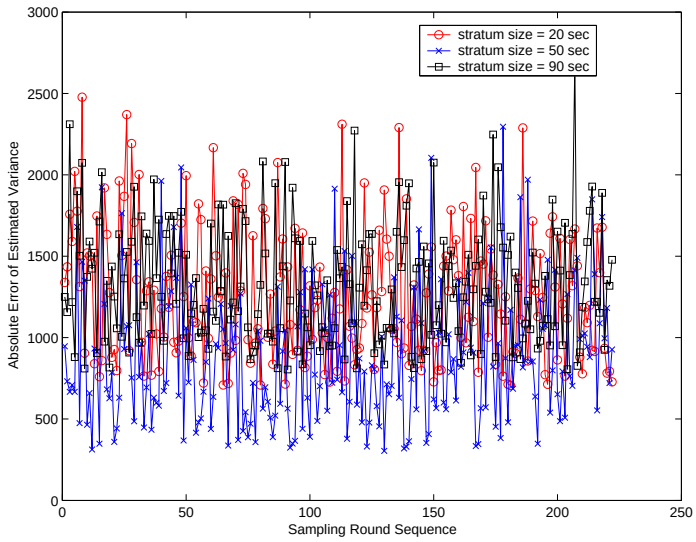


Figure 5.10 Absolute Error of Estimated Variance of adaptive stratified sampling with stratum size = 20, 50 and 90 seconds.

5.4 Simulation Results

The simulation results of the proposed adaptive sampling with the stratum size of 50 seconds are compared with the simulation results of the timer-based systematic sampling, timer-based Poisson sampling and stratified sampling with optimum allocation. The comparisons are shown in Table 5.2, Figure 5.11 and 5.12.

Table 5.2
Main simulation results of sampling tests with different sampling methods. Stratum size: 50 seconds (true values are: $\mu = 86.824$ ms, $\sigma^2 = 8529$)

Sampling method	M	$AMean = \frac{1}{M}\sum \hat{\mu}_i$	$AVar = \frac{1}{M}\sum s_i^2$	$MSE = \frac{1}{M}\sum(\hat{\mu}_i - \mu)^2$
Systematic	222	74.803 ms	5996	145
Poisson	222	64.998 ms	4710	478
Stratified with optimum allocation	222	88.023 ms	8959	5
Adaptive stratified	222	84.895 ms	8081	10

From the simulation results, we can reach two conclusions. Firstly, the proposed adaptive stratified sampling scheme produces a better performance than the timer-

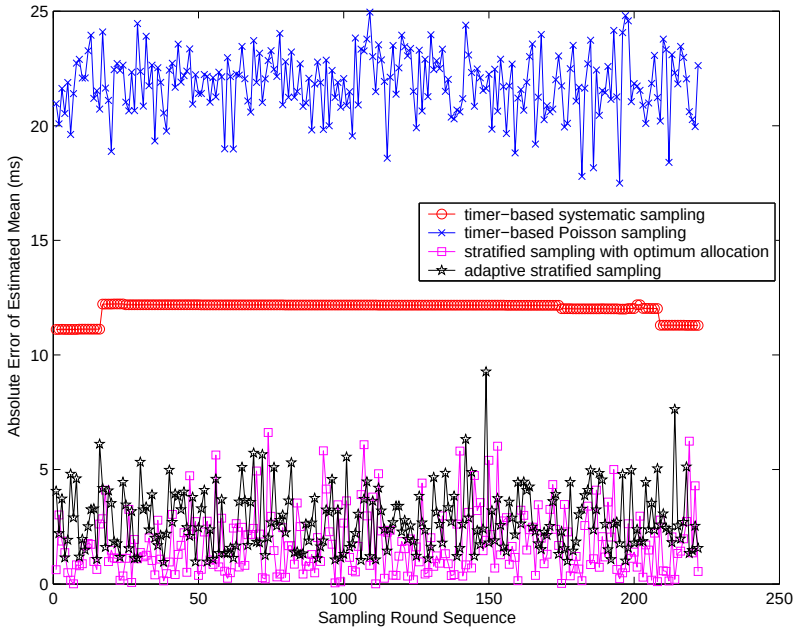


Figure 5.11 Comparison of Absolute Error of Estimated Mean for different sampling methods. Stratum size: 50 seconds, sampling rounds: 222.

based systematic sampling and the timer-based Poisson sampling. Secondly, the proposed adaptive stratified sampling achieves approximately the same performance as the stratified sampling with optimum allocation.

5.5 Summary

In this chapter, a novel adaptive stratified sampling method was proposed in order to address the challenges of implementing the stratified sampling with optimum allocation and to obtain a higher accuracy of estimate. Firstly, a brief introduction to the LMS algorithm was provided. Secondly, a detailed description of the adaptive stratified sampling algorithm was presented. A fourth-order LMS algorithm was employed to predict the standard deviation of packet delay, which was used to compute the sample size $\hat{n}_l$. Thirdly, a theoretical analysis on the prediction error was presented. The relative error between the variance of the sample mean of adaptive stratified sampling and the variance of the sample mean of stratified sampling with optimum allocation was related to the ratio of $\hat{n}_l/n_l$. Fourthly, a discussion of the

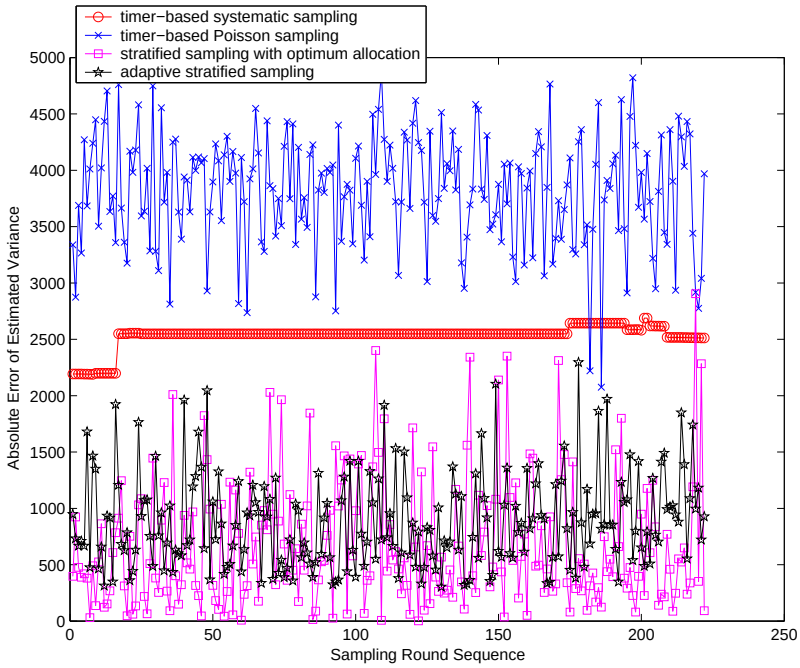


Figure 5.12 Comparison of Absolute Error of Estimated Variance for different sampling methods. Stratum size: 50 seconds, sampling rounds: 222.

construction of strata was presented. The stratum size was determined using the autocorrelation of packet delay. Simulations with a stratum size of 20 seconds, 50 seconds and 90 seconds were deployed respectively, among which the stratum size of 50 seconds produced the best performance. Finally, performance comparisons of different sampling methods were presented. The adaptive stratified sampling produced approximately the same performance as the stratified sampling with optimum allocation.

Chapter 6

Monitoring Software Design

6.1 Introduction

In this chapter, I introduce the software design. Firstly, I introduce the software environment and the functionality of the software. Secondly, the software designs on measurements using ICMP (Internet Control Message Protocol), UDP (User Datagram Protocol) and TCP (Transmission Control Protocol) protocols are presented. ICMP measurement provides a simple idea of network performance and does not require the server program; UDP measurement can be used to obtain the QoS of those applications running on the top of UDP (e.g., VoIP); TCP measurement can be used to obtain the QoS of those applications running on the top of TCP (e.g., HTTP, FTP, web service). Thirdly, the discussion on how to obtain an accurate time in the Windows operating system for delay measurements is presented. Systematic error in delay measurements, which is caused by packet processing in both the client and server, is calibrated and measurement results in different network environments are presented. The thread structure used on both the client side and the server side of the software is also presented. The software is provided in the attached CD.

6.2 Software Environment

The QoS monitoring software is written in C++ language and is developed with Microsoft Visual Studio 6.0. The measurements can be taken using the TCP, UDP or

ICMP protocol. The software uses a client-server architecture. The client initiates the measurement request, the server responds to the client's request, performs some simple computations and returns the results to the client. Figure 6.1 illustrates the client-server architecture. The server can support measurements of up to 60 TCP clients and up to 60 UDP clients at the same time. The maximum number of ICMP clients is determined by the limit imposed by the operating system of the server. However, too many client connections to the same server may create a "bottleneck" effect at the server, which may significantly slow down the response speed of the server and increase traffic congestion in the server network. A rule of thumb is that the bandwidth consumed by the measurement traffic should be smaller than 5% of the total bandwidth of the server.

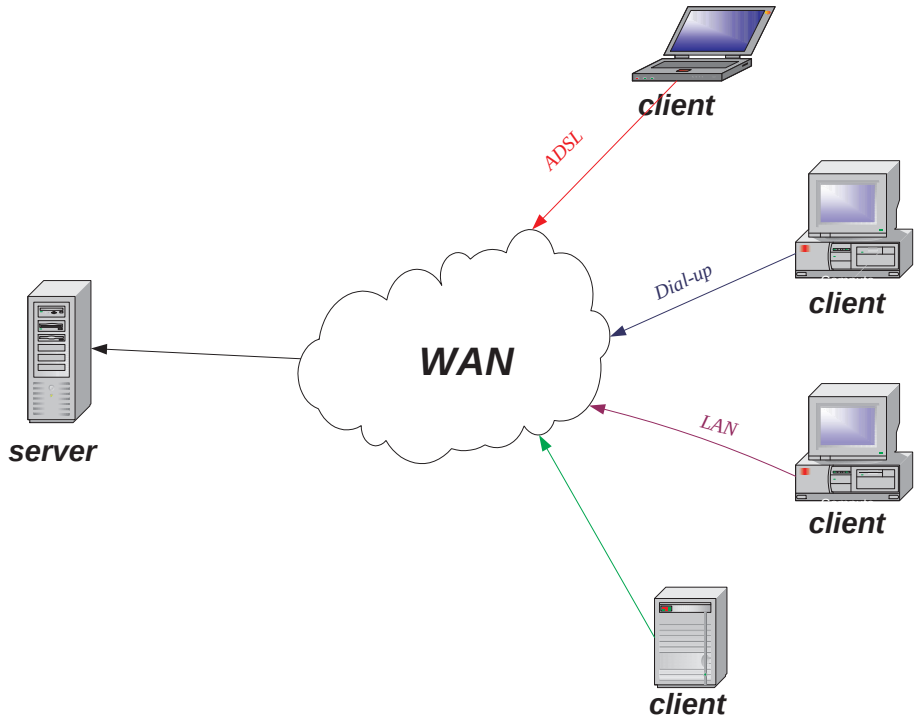


Figure 6.1 Client-server architecture

6.2.1 IP Precedence Setting

The software is designed to support measurements of the QoS of different traffic classes. Network traffic is classified into multiple classes using the "IP Precedence" field in the IP packet header. The "IP Precedence" field is a subfield of the "Type of

Service" field in the IP packet header. Figure 6.2 shows the structure of the IP packet header [46]. The "Type of Service" field is one octet long and consists of three subfields, as shown in Figure 6.3. The first subfield, labelled "PRECEDENCE", is intended to denote the importance or priority of the IP packet. The second subfield, labelled "TOS", denotes how the network should make tradeoffs between throughput, delay, reliability, and cost. The last subfield, labelled "MBZ" (must be zero), is currently unused [47] (RFC 1349).

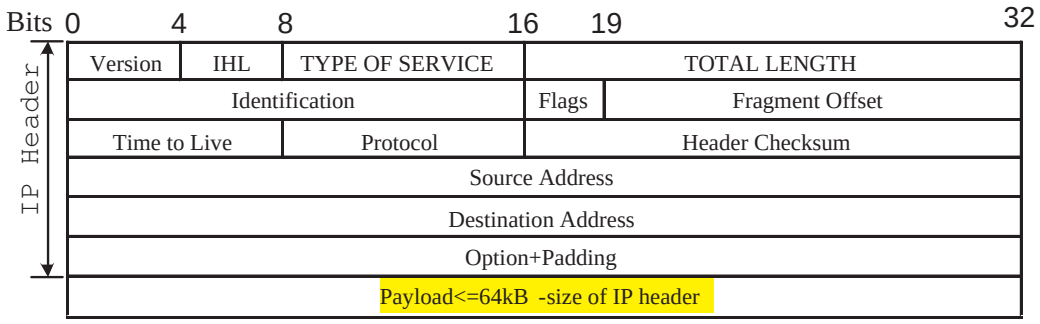


Figure 6.2 IP header

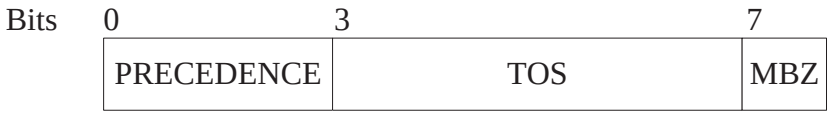


Figure 6.3 ToS field in IP header

Optus classifies traffic into four classes based on the "IP Precedence" field. The value of the "IP Precedence" is between 0 and 7. The mapping between the "IP Precedence" field and traffic classes is shown in the following.

- gold-rt: IP-Precedence=5;
- gold-nrt: IP-Precedence=4;
- silver: IP-Precedence=3;
- bronze: IP-Precedence=0,1,2.

In the software, I specify the traffic class to be measured by changing the value of the “IP Precedence” field of the IP packets generated by the measurement program. It should be noted that the “IP Precedence” field is meaningful only when the user has indeed subscribed to the service corresponding to the value indicated in the “IP Precedence” field and the field is not changed by other user software and/or by the ISP.

6.2.2 Software Platform

The software is designed to operate in most versions of the Windows operating system. These include Windows 98, ME, 2000 and XP. However, due to incompatibility in the network programming interface, the program cannot run under Windows 95 and CE. The software has passed the tests under Windows 98, 2000 and XP. In addition, the program only supports IPv4 due to the reason that most Win32 platforms including Win2000 do not support the IPv6 network stack [48].

6.2.3 Network Programming Interface

The software communicates with the network transport layer using Windows Socket. Windows Socket is a network programming interface and supports multiple network protocols. It was developed from the Berkeley Sockets Interface and further enhanced with a group of extended functions for Windows [49]. Because of its application in Windows platforms, Windows Socket has become more and more popular. Now it is the most important network programming interface in Windows platforms.

There are two main versions of Windows Socket: Winsock 1.1 and Winsock 2.2. Winsock 1.1 has better compatibility with Windows operating systems. Winsock 2.2 is a recent version and has been enhanced with new and more efficient network functions [50]. We have chosen Winsock 2.2 as the programming interface for our program.

6.3 Software Functionality

This software uses active sampling to perform QoS measurements and it supports measurements using TCP, UDP and ICMP protocols. The following shows the metrics measured under the three protocols:

- TCP: RTT, one-way jitter (i.e., from client to server) and packet loss. As TCP is a reliable and connection-oriented transport layer protocol, it may hide packet loss from the upper layers. TCP packet loss is measured indirectly using Karn's algorithm and Jacobson's algorithm [51] (RFC 2581), [52] (RFC 2988). Karn's algorithm and Jacobson's algorithm will be introduced later in Section 6.4.3.
- UDP: RTT delay, one-way jitter, one-way loss (i.e., from client to server);
- ICMP: RTT delay, round-trip jitter and round-trip loss. This is similar as the conventional "ping" program. As determined by the ICMP protocol, ICMP based measurements can only measure round-trip metric.
- network availability: the software also provides a simple idea about the network availability based on the TCP, UDP or ICMP measurement. It diagnoses and records the time when the network is available and when the network is unreachable.

Users can specify the sampling method (i.e., periodic, Poisson or stratified sampling), sampling frequency, packet size and packet size distribution (i.e., constant or random size). Sampling is timer-based rather than count-based. That is, the generation of sampling packets is triggered by a timer. The support of many sampling techniques is due to the concern that the user may want to compare our program with some legacy softwares, which typically use periodic sampling or Poisson sampling. It is expected that in different environments (e.g., dial-up internet, ADSL, LAN), different sampling parameters should be used for the most accurate measurements. The flexibility in choosing sampling parameters allows the program to be used in different environments.

6.4 Measurement Using ICMP, UDP and TCP Protocols

In the following paragraphs, the software design for ICMP, UDP and TCP measurements are presented.

6.4.1 ICMP Measurement

ICMP is defined in [53] (RFC1256) and is part of the TCP/IP protocol suite. It is used by hosts, routers and gateways for a variety of functions, especially for network management. The network performance measurement is one of the applications of ICMP [46, pp. 386]. ICMP-based measurement tools (e.g., ping) are easy to use and they have been used to measure the RTT delay and packet loss. But the results they produce are limited and inaccurate [54]. For example, common ICMP-based tools send probing packets to a host and measure loss by observing whether or not response packets arrive within a specified time interval. Thereby we obtain the round-trip loss, not the one-way loss. Furthermore, there are two problems with this approach:

- Asymmetric loss: If the forward path (i.e., from client to server) and the reverse path are symmetric, we can obtain an estimate of the loss rate using the following equations, assuming packet losses in the forward path and in the reverse path are independent:

$$loss_{round-trip} = 1 - \frac{number_{received}}{number_{sent}} = 1 - ((1 - loss_{forward}) \cdot (1 - loss_{reverse})), \tag{6.1}$$

$$loss_{forward} = loss_{reverse}. \tag{6.2}$$

  Then we can calculate the packet loss ratio in the forward path (reverse path), $loss_{forward}$ ($loss_{reverse}$), through Equation 6.1 and Equation 6.2. Unfortunately, in most case the forward path and the reverse path are asymmetric, i.e., $loss_{forward} \neq loss_{reverse}$. So we can only obtain the round-trip loss, not the one-way loss.

- ICMP filtering: ICMP-based tools rely on the deployment of ICMP echo or ICMP time-exceeded messages to coerce response packets from the destination host [54]. However, this function of ICMP is sometimes used for malicious purpose, e.g., denial of service attack. For security reasons, many networks (e.g., microsoft.com) filter out ICMP packets. Therefore, this has limited the use of ICMP as a measurement protocol.

Still another feature impeding the use of ICMP for measurement is that ICMP packets are often considered by the network as control packets and treated differently from the user traffic. Therefore, they may not be able to accurately measure the QoS of ordinary user traffic. This program measures ICMP RTT, round-trip loss and round-trip delay variation.

6.4.2 UDP Measurement

UDP is a connectionless and unreliable transport layer protocol: each output operation by a process produces exactly one UDP datagram, which causes one IP packet to be sent [55].

Based on UDP, our program provides RTT delay, one-way jitter and one-way loss measurements. As an unreliable protocol, UDP allow us to measure the one-way packet loss much easier and more accurately than the reliable protocols (i.e., TCP) do. The client sends a group of probing packets and UDP will not retransmit a packet if the packet is lost. Meanwhile, the server counts the number of probing packets received and echoes the received packets with some test information (e.g., total number of packets received by the server). Thus we can obtain three parameters: Num_{sent} (number of packets sent), Num_{recv} (number of packets received by the server) and Num_{echo} (number of packets received by the client from the server), and then we can calculate the one-way loss: $loss_{forward}$, $loss_{reverse}$. Figure 6.4 shows their relationship.

$$loss_{forward} = 1 - \frac{Num_{recv}}{Num_{sent}}, \tag{6.3}$$

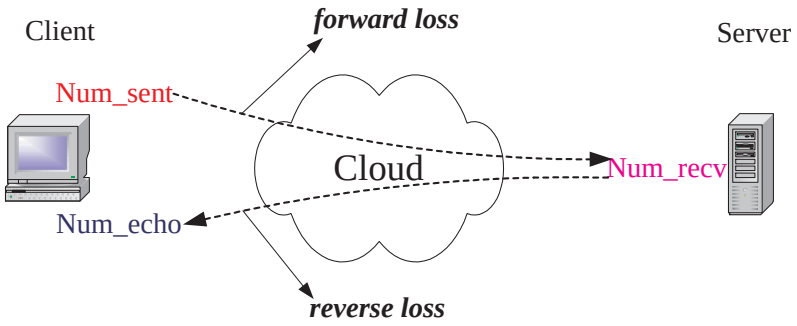


Figure 6.4 One-way loss

$$loss_{reverse} = 1 - \frac{Num_{echo}}{Num_{recv}}. \tag{6.4}$$

Figure 6.5 shows the design of the UDP service data unit (SDU). The same SDU is used by both the server and the client for simplicity. The UDP SDU contains the following fields:

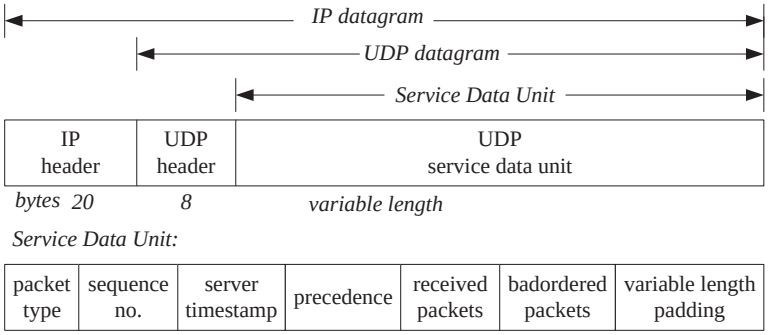


Figure 6.5 UDP encapsulation

- Field “packet type”: represents the type of UDP packet that being sent, e.g., “initial-test” packet, probing packet, or “end-test” packet;
- Field “sequence number”: represents the sequence number of the packet from the client side;
- Field “server timestamp”: represents the time when this packet is received by the server;

- Field “precedence”: represents the IP Precedence specified by the user;
- Field “received packets”: represents the total number of packets received by the server;
- Field “bad-ordered packets”: is a 32-bit number, the high 8-bit represents the status of this packet (whether it is bad-ordered or not), the low 24-bit represents the total number of bad-ordered packets.
- Field “padding”: is a variable length field and is used to pad the packet length to that specified by the user.

Figure 6.6 shows the flowchart of the UDP measurement program. Firstly, the client sends an “initial-test” packet to request the server to initialise parameters for a new test. When the client receives the “OK” packet from the server, it starts sending a group of probing packets to the server with specified packet size and time interval. On reception of a probing packet from the client, the server sets the related test information (e.g., receiving time, total number of received packets, total number of bad-ordered packets and status of the packet) into the corresponding fields of the packet SDU and returns the packet as soon as possible. The client records the time when the packet is sent, denoted by $sending\_time$. Every time the client receives the response packet from the server, it records the time when the response packet is received, denoted by $echoing\_time$, the time when the server receives the probing packet, denoted by $server\_timestamp$, the total number of packets received by the server, which is counted by the server and the total number of bad-ordered packets counted by the server. The RTT delay, one-way jitter and one-way loss metrics are calculated with every N packets being sent. During the test, data files are locked for data integrity and the user will be able to observe measurement results after the test.

At the end of the test, the client sends a “test-end” packet to the server to complete the session, saves all useful information to hard disk and cleans up buffers and memories.

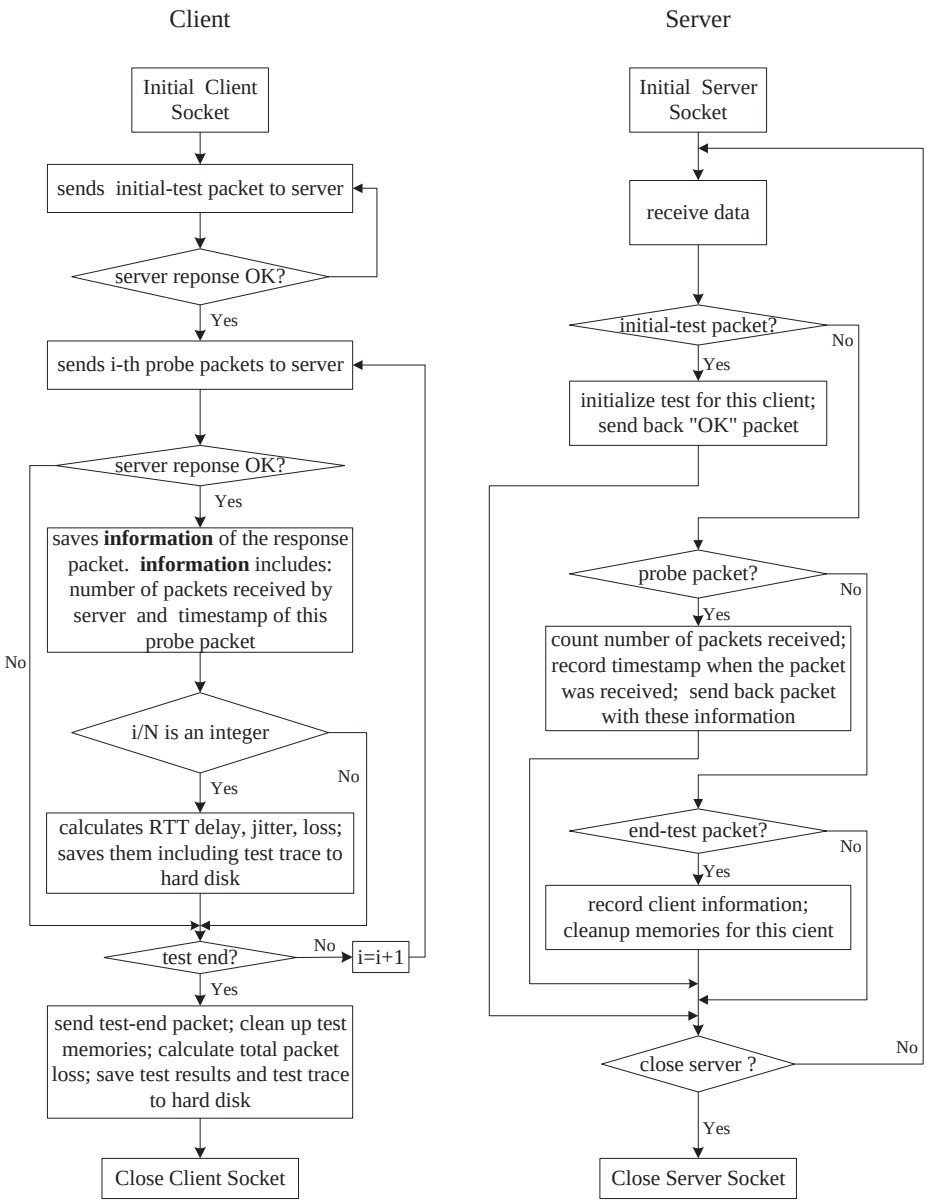


Figure 6.6 Flowchart of UDP measurement program

On the server side, the server cleans up data related to the test of this client when it receives the “test-end” packet.

Because UDP is connectionless, there is no connection establishment phase when using UDP. However, this characteristic causes some problems. As we have discussed, the server needs to count the number of packets it receives and record the receiving time of every received packet for measuring the one-way loss and one-way jitter. So

we must maintain state information for each test session. In normal situation, the client sends a “test-end” packet to inform the server that the test is over. But if some unexpected errors occur (e.g., client computer shuts down unexpectedly or the “test-end” packet is not received successfully), the server has no way of knowing this. Consequently, the server will maintain the state information indefinitely, which may consume server resources. To solve this problem, the server only maintains a “soft” state for each test session. If there is no packet coming within a pre-determined amount of time, the server will consider that the client has left and clean up the state information and release resources reserved for the test session.

6.4.3 TCP Measurement

TCP provides a different service to the application layer from UDP, even though TCP and UDP use the same network layer (IP). TCP provides a connection-oriented, reliable and byte stream service. The term “connection-oriented” means two applications using TCP must establish a TCP connection with each other before they can exchange data [56].

For a byte stream service, there are no record markers automatically inserted by TCP [56]. For example, if the application on one end sends 32 bytes, followed by a packet of 64 bytes, then followed by a packet of 128 bytes, the application at the other end of the connection cannot tell the size of the individual packet. The other end may read the total 224 bytes at one time. Figure 6.7 shows the UDP receiving buffer and TCP receiving buffer when transmission occurs.

Therefore, the properties of the byte stream service may cause problems when there are consecutive packets in the receiving buffer of server or client. If the server reads two or more packets together, it would only respond one packet back to the client and the others are lost by mistake. The same thing occurs on the client side. To solve the problem, I add a data-field storing packet size into the TCP SUD. Figure 6.8 shows the design of the TCP service data unit (SDU). The data fields have the same meaning as the UDP SDU. Now, each TCP packet carries its own size (in bytes). When receiving a packet, the server or client compares the buffer length[1] and packet

[1]Here the buffer is defined in the program and is used to store received data from the TCP buffer.

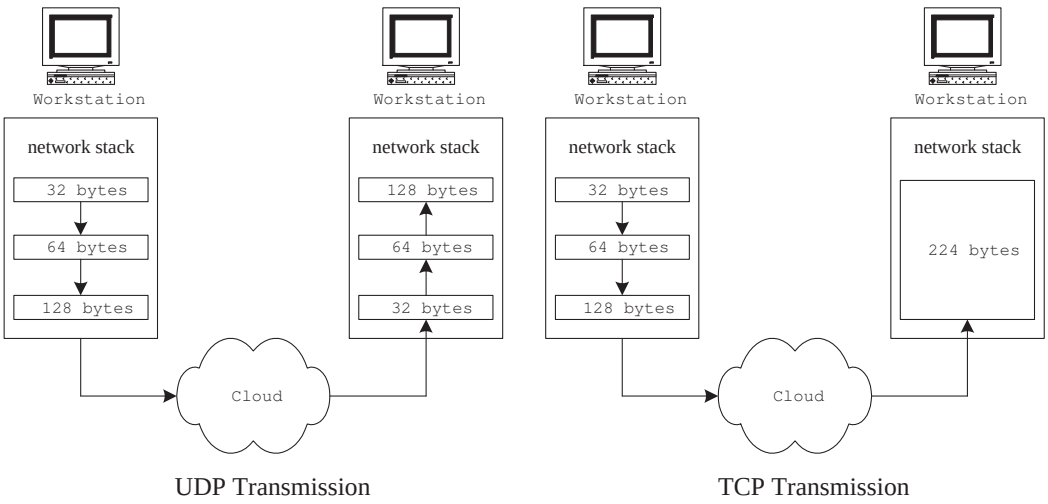


Figure 6.7 UDP receiving buffer and TCP receiving buffer

size it has just read from the buffer. If the buffer length is greater than the packet size, it means that there are more than one packets in the buffer. In this case, the server or client moves the buffer pointer[2] from the current position of the buffer by that packet size and starts reading another packet in the buffer, and so forth. Figure 6.9 displays the data-reading procedure in the TCP measurements.

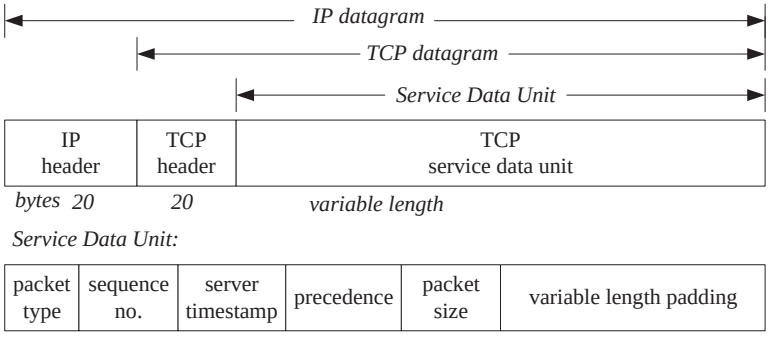


Figure 6.8 TCP encapsulation

Based on TCP, our program provides RTT delay and one-way jitter measurements and an estimated value for packet loss. The calculation of RTT and jitter is the same as that used in UDP measurements, except that TCP retransmission must be

[2]Here the buffer pointer is defined in the program, representing the position where the host should start to read data in the buffer.

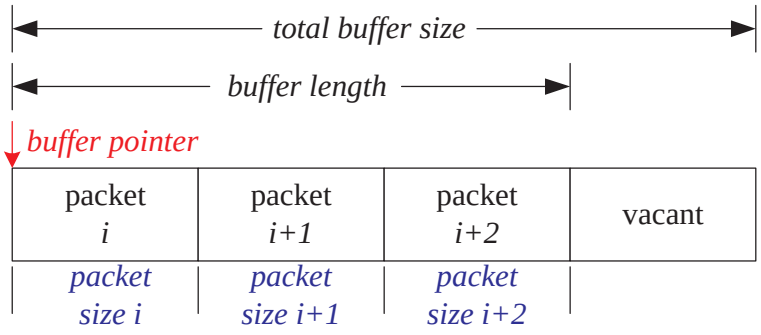


Figure 6.9 Data-reading procedure in the TCP measurement

considered in calculating TCP RTT. The test procedure is listed in the following:

- Establish a TCP connection between the client and the server;
- The client sends an “initial-test” packet to the server and waits for a response;
- After receiving a response from the server, the client sends probing packets with specified packet size and interval;
- At the end of the test, the client sends an “end-test” packet to the server, saves the test information to hard disk and cleans up the buffers and memories of the systems. On the server side, the server cleans up resources reserved for the test of this client when it receives the “end-test” packet.

Figure 6.10 shows the flowchart of the TCP measurement program.

The major challenge in the TCP measurements is that TCP is a reliable protocol and TCP packet retransmission may hide packet loss from the application layer. This creates difficulty for packet loss measurements using TCP. To solve the problem, we implement the Karn’s algorithm and the Jacobson’s algorithm [51] (RFC 2581), [52] (RFC 2988) in our program to infer TCP packet loss from the RTT measurements. According to the IETF standard [51] (RFC 2581), [52] (RFC 2988), a TCP host must implement Karn’s algorithm and Jacobson’s algorithm for computing the retransmission timeout (RTO).

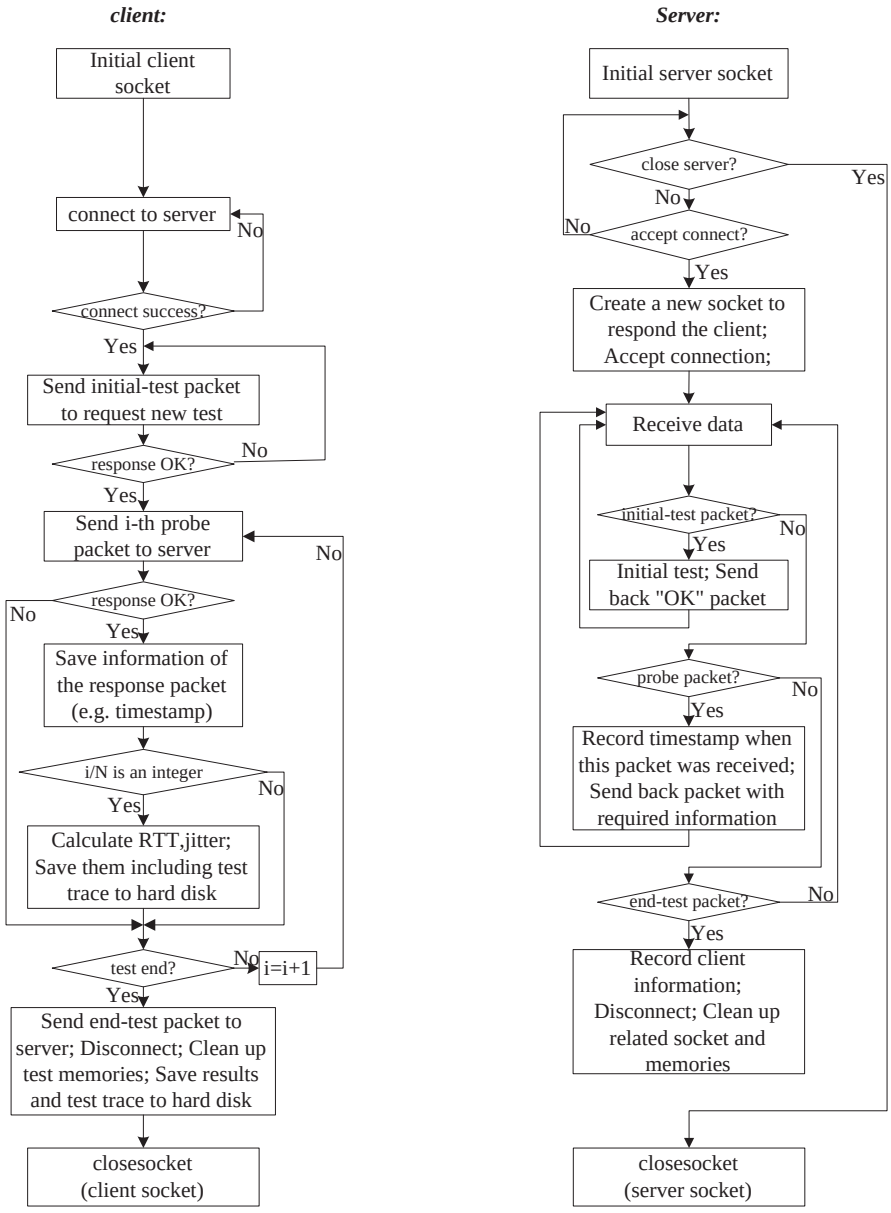


Figure 6.10 Flowchart of TCP measurement program

- Jacobson's algorithm is used for computing the smoothed round-trip time, including a simple measure of the variance;
- Karn's algorithm is used for selecting RTT measurements to ensure that ambiguous round-trip time will not corrupt calculation of the smoothed round-trip time.

Packets whose RTT exceeds RTO will be deemed as packet loss by TCP protocol and will be retransmitted.

Jacobson's algorithm defines a smoothed RTT (SRTT), which is updated with each RTT sample through the equation:

$$SRTT_i = SRTT_{i-1} + (\alpha \times Error_i), \tag{6.5}$$

where $Error_i$ is defined as the difference between the i^{th} RTT sample and the earlier estimated SRTT:

$$Error_i = RTT_i - SRTT_{i-1}, \tag{6.6}$$

and α is a constant, which is chosen to be 0.125. The RTT deviation is estimated by:

$$Dev_i = Dev_{i-1} + \delta(|Error_i| - Dev_{i-1}), \tag{6.7}$$

where δ is a constant and is equal to 0.25.

Finally, the RTO_{i+1} for the $(i+1)^{th}$ packet is calculated as:

$$RTO_{i+1} = SRTT_i + (\beta \times Dev_i), \tag{6.8}$$

where β is a constant, which is equal to 4.0. The initial values of these parameters are: $RTO_1 = 3$, $SRTT_1 = RTT_1$, $Dev_1 = RTT_1/2$, $RTO_2 = SRTT_1$. Karn's algorithm specifies that when computing the RTT estimate (i.e., SRTT), samples corresponding to retransmitted segments are ignored, and RTO is doubled with each retransmission.

Using the two algorithms, I have emulated the TCP congestion control process in our program. If a packet's RTT exceeds the corresponding RTO, that packet is considered as a packet that has been lost and retransmitted.

6.5 Accuracy of Time Measurement

Measurements of RTT and one-way jitter rely on accurate value of the sending time of the packet at the client end and the receiving time of the packet at both the client and the server ends. Moreover, specification of packet inter-arrival time also needs

accurate time. Consequently, obtaining an accurate time is important for the measurement software. It is well known that the Windows operating system is based on a "message mechanism". If CPU is taken up by a high-priority process, or the system resource is insufficient, the message in the queue will be temporarily locked and delayed. Therefore, it is very difficult to obtain an accurate time by using the conventional Message Triggering Timer (e.g., WM_TIMER) or Timers derived from Message Triggering Timer.

Through the literature review, we learned that there are at least seven methods for obtaining time with different accuracy [57]:

- WM_TIMER message mapping: low resolution ($\geq 30ms$) with a low priority in multi-thread operating systems;
- Sleep() function: low resolution ($\geq 30ms$) and the thread can not process other tasks when it is in the "Sleep" state;
- COleDateTime and COleDateTimeSpan: low resolution;
- GetTickCount(): short period timing resolution $\geq 15ms$, long period timing resolution $\geq 50ms$;
- Multimedia Timer timeGetTime(): a timer provided by Microsoft for multimedia application. Resolution is in the order of ms;
- Multimedia Timer timeSetEvent(): a timer provided by Microsoft for multimedia application. Resolution is in the order of ms;
- QueryPerformanceCounter and QueryPerformanceFrequency: the highest resolution timer in Windows Platforms (Win 95 and later versions) can be accessed by Windows API functions supported by Microsoft Visual C++ . Resolution is dependent on the hardware and the CPU clock and is typically in the order of μs.

As our software requires accurate time measurements, we have chosen "QueryPerformanceCounter" and "QueryPerformanceFrequency" for delay and jitter measurements.

Another major error source in measuring RTT is the time spent by the server in receiving and recognising the packet from the client, and then producing and sending the corresponding response, and the time required at the client end to generate the sampling packets and to process the response packet from the server. This error depends on the hardware of both the client and the server, the number of concurrent test sessions at the server end and other software running on the client and the server. To calibrate the magnitude of this error, several one-hour test sessions are conducted between two Pentium IV computers connected to the same 100-Mbps LAN. During the test, there are other users using the LAN. A TCP test session, a UDP test session and an ICMP test session were conducted at the same time between the two computers. The test parameters are: sampling interval 1 s, random sampling, packet size constant, 100 bytes. Figure 6.11, 6.12 and 6.13 show the cumulative distribution function of RTT using TCP, UDP and ICMP respectively. The average RTTs are 0.333 ms, 0.345 ms and 0.133 ms. This indicates that the aforementioned systemic error is upper bounded by 0.35 ms, which is considered to be trivial and can be ignored. The average RTT of TCP measurements and UDP measurements is higher than the average RTT of ICMP measurements by 0.2 ms. This may be caused by the fact that ICMP protocol has been embedded into the operating system. Therefore the server can respond more quickely to the ICMP packets. For TCP and UDP packets, additional handling is required by the server program. It is worth noting that no packet loss has been observed in TCP and UDP measurements, however, in ICMP measurements, a packet loss ratio of 0.0281% (corresponding to one packet loss) has been observed. The same pattern (i.e., TCP and UDP measurements have no loss and UDP measurements sometimes have significant loss even when measurements are conducted in exactly the same environment) is also observed in the other network environments. It is considered that in this case, ICMP loss is unlikely to have been caused by the network and it may be attributed to problem, in the NIC (network interface card) or operating system of the computer.

As a reference, Figure 6.14, 6.15 and 6.16 show RTT measurements performed in a number of different environments. In particular, the RTT measurement results shown in Figure 6.16 between a computer located at the University of Sydney and a computer located at NICTA in Canberra are impressively good. NICTA in Canberra

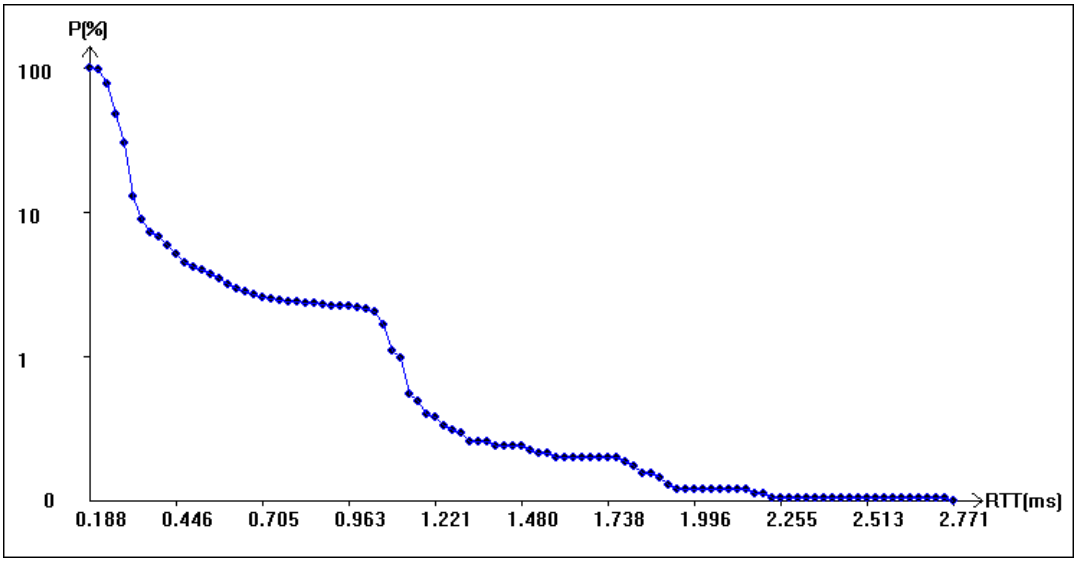


Figure 6.11 RTT measurements using TCP protocol between two computers connected to the same LAN at the University of Sydney. The average RTT is 0.333 ms.

subscribed to a premium Internet service and has transited to VoIP services.

6.6 Multi-thread and File Management

The software is written using multi-thread. The multi-thread server program can achieve better performance in a high-speed computer, especially on a multi-CPU computer.

In the Windows operating system, there are two types of thread: the UI (User Interface) thread and the Assistant thread (or working thread) [57], [58]. On the server side, the primary thread is the UI thread and processes most jobs (e.g., Packet transmission, GUI). In addition, there is a working thread which is used to synchronously save test information. This working thread is started together with the main server program and is terminated when the server program is stopped by the user.

On the client side, the primary thread is the UI thread and mainly process GUI. In addition, there are four more working threads:

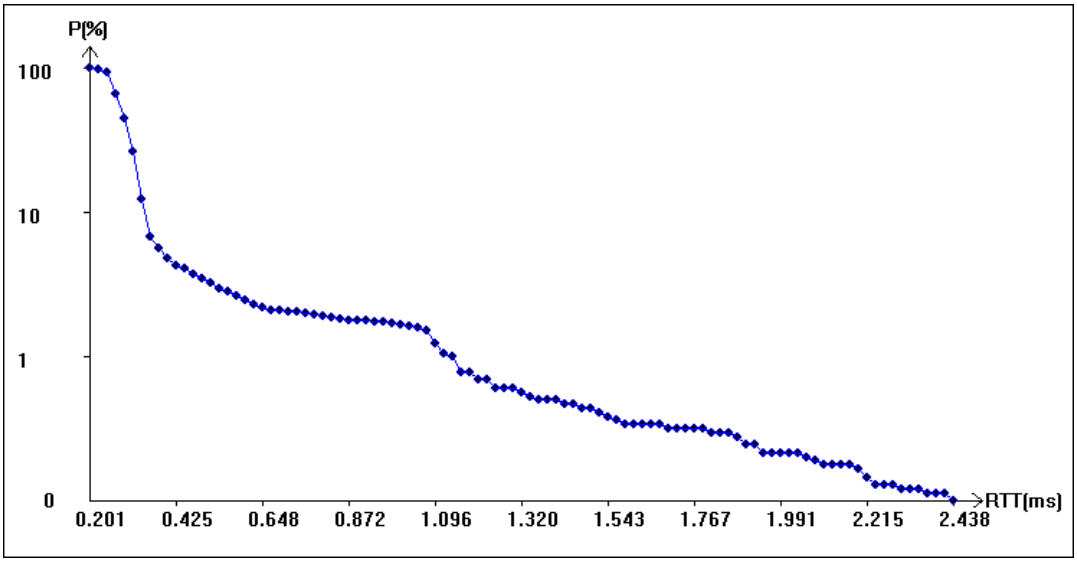


Figure 6.12 RTT measurements using UDP protocol between two computers connected to the same LAN at the University of Sydney. The average RTT is 0.345 ms.

- TCP thread: processes TCP measurements;
- UDP thread: processes UDP measurements;
- ICMP thread: processes ICMP measurements;
- Saving thread: periodically saves measurement information and error messages to the hard disk.

The first three working threads are independent from each other, so the user can run TCP measurements, UDP measurements and ICMP measurements independently at the same time. The last thread periodically saves measurement information and error messages to the hard disk.

Memory leak is a major problem in designing multi-thread programs [58]. If memory leak occurs in the program, especially in the server program, it will continuously consume system resources until the computer is stopped due to lack of resources. We have carried out a lot of debugging tests and macro diagnosis to ensure that our program has no memory leak.

Another challenge in designing the software is file management. Both the server

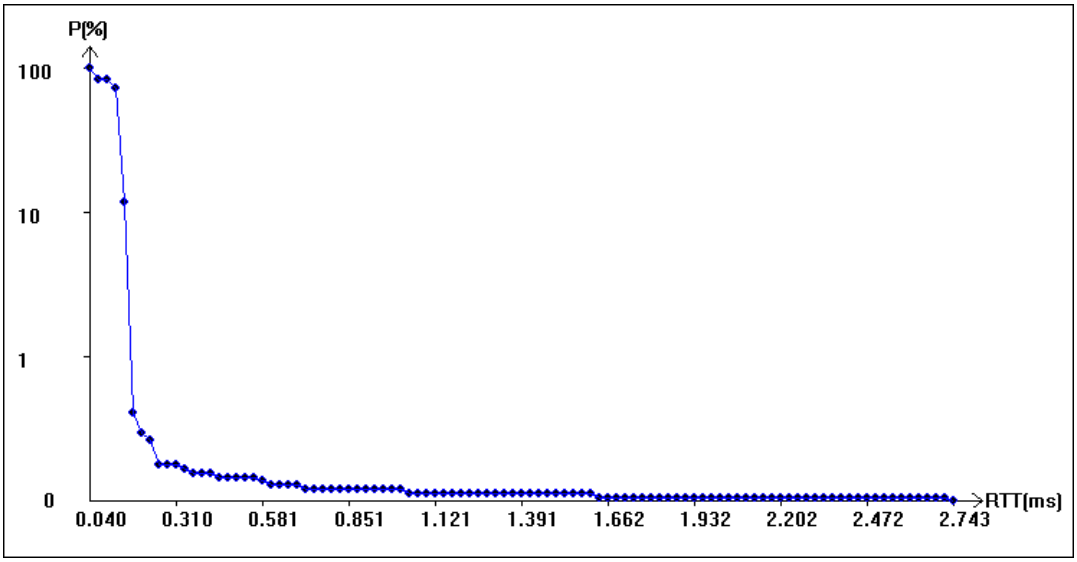


Figure 6.13 RTT measurements using ICMP protocol between two computers connected to the same LAN at the University of Sydney. The average RTT is 0.133 ms.

and the client programs have functions that save useful data to hard disk (e.g., client information, test results, error messages, etc.). A user can observe these data at any time except when the saving thread performs the file I/O operation, during which files are locked to ensure integrity of data. At the end of the test, the user can save these files to other directories for later use.

6.7 GUI Design

In this section, I shall briefly introduce major components in the server GUI and the client GUI.

There are five major components in the server GUI; each is responsible for one function:

- configuration. It is responsible for configuring server parameters, e.g., port number to listen on;
- server operation, e.g., starts and stops the server;

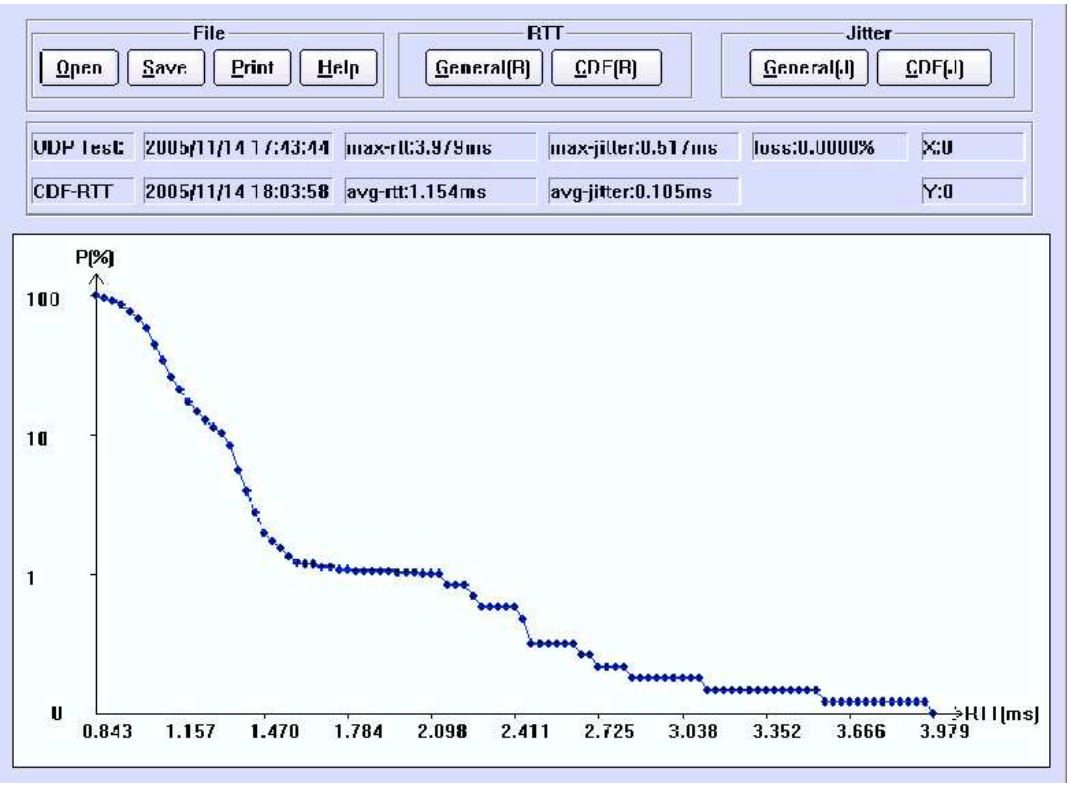


Figure 6.14 RTT measurements using UDP protocol between a server located at the University of Sydney and a client located at NICTA at ATP, Sydney. Both the client and the server are connected to a high-speed LAN, then to a WAN. The average RTT measured is 1.154 ms.

- online client information display, including display of online clients' IP address, Port, protocol, test start date and time;
- test records display, including clients' IP address, protocol, test start date, test start time, test end date and test end time;
- error messages and help information display. This includes error date, error time, error reasons and help documentation.

There are six major components in the client GUI:

- configuration, e.g., server IP address, server port, packet size, packet size distribution, sampling methods, sampling frequency, IP Precedence, test time;

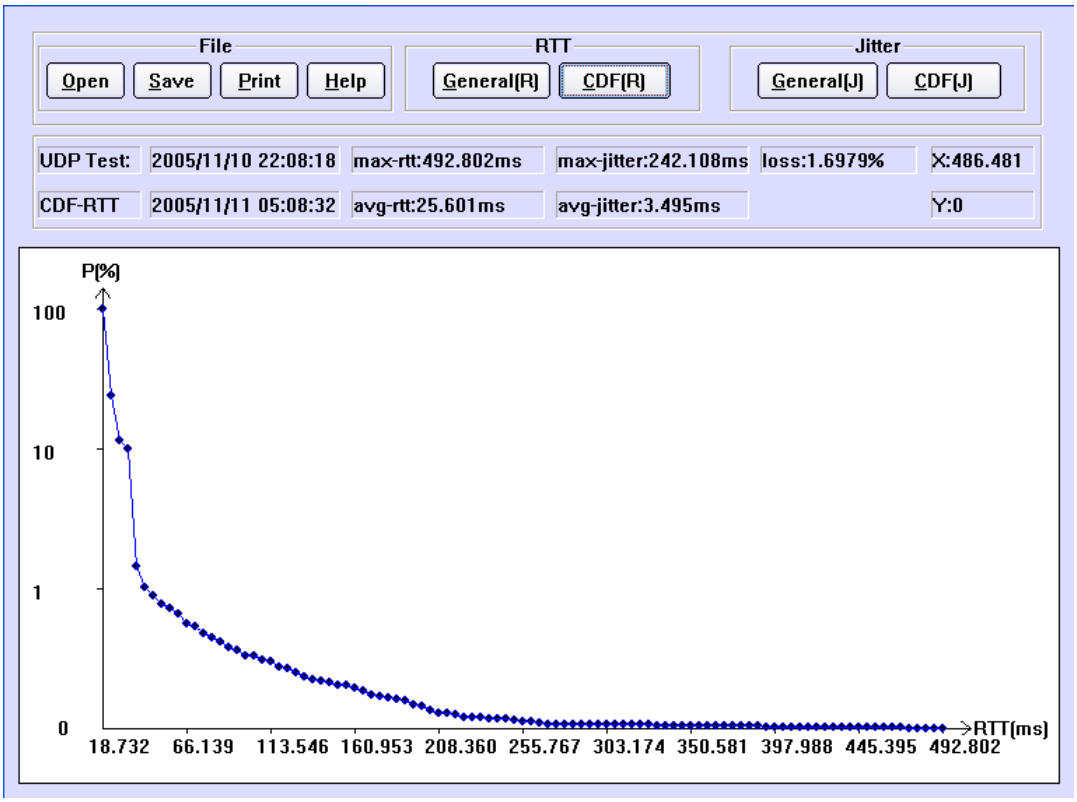


Figure 6.15 RTT measurements using UDP protocol between a server located at the University of Sydney and a client located at Carlton, Sydney. The server is connected to a WAN through a high-speed LAN. The client is connected to a WAN through a wireless LAN, then ADSL. The average RTT measured is 25.601 ms.

- test operation: starts TCP test, stops TCP test, starts UDP test, stops UDP test, starts ICMP test, stops ICMP test;
- test progress display;
- test result display. This includes display of overall test results and individual test packet information.
  - overall result display: num_sent, num_loss, num_echo, max_delay, min_delay, mean_delay, max_jitter, min_jitter, mean_jitter, test start date, test start time, test end date, test end time, operating system information, protocol;
  - packet information display: display of individual packet information, e.g., packet sequence number, sending date, sending time, RTT delay, jitter, packet size.

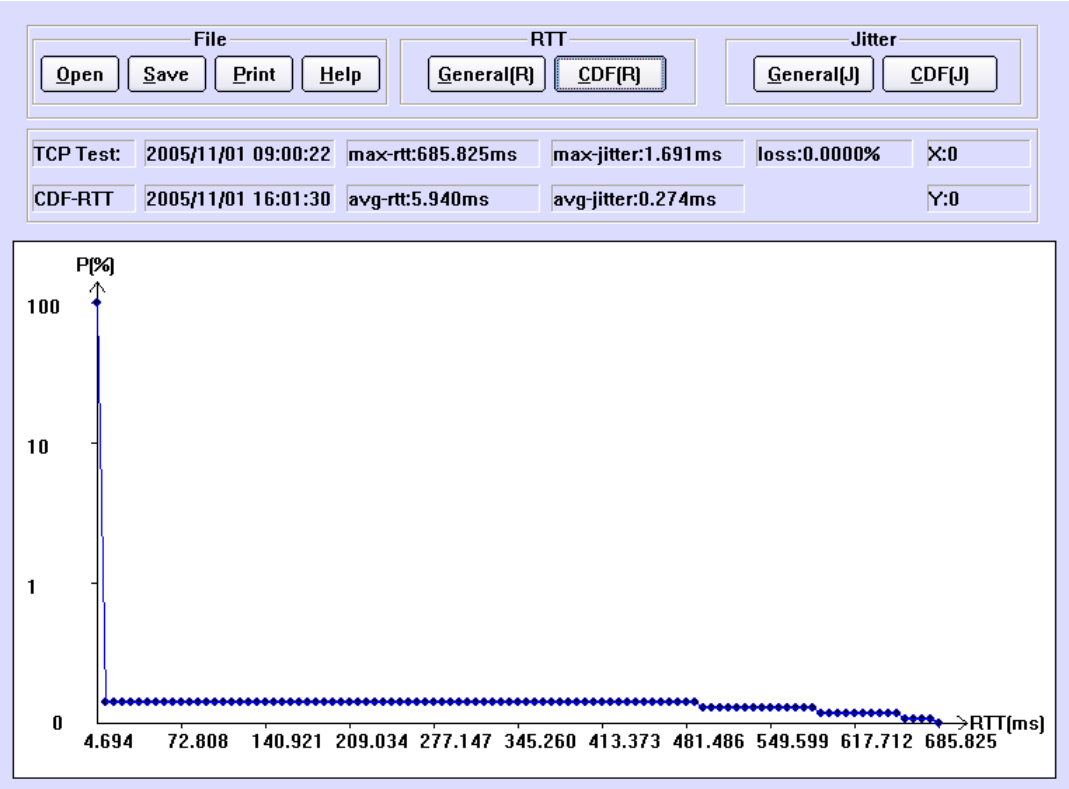


Figure 6.16 RTT measurements using TCP protocol between a server located at the University of Sydney and a client located at NICTA in Canberra. Both the client and the server are connected to a high-speed LAN, then to a WAN. The average RTT measured is 5.940 ms.

- graphical display of test result;
- error messages and help information display.

More detailed information about the software can be found through the help menu in the software.

6.8 Summary

This chapter described the software design. Firstly, an introduction to the software environment and the software functionality was presented. Secondly, the designs of the ICMP measurement, UDP measurement and TCP measurement in this software were presented. Thirdly, a discussion on how to obtain an accurate time measure-

ments in the Windows operating systems was provided. Fourthly, systematic error in delay measurements was calibrated, and the systematic error was bounded by 0.35 ms, which was trivial and could be ignored. Finally, measurement results in real network, as well as the GUI design of the software, were presented.

Chapter 7

Conclusion

In this chapter, I summarize my work on the project "A Quality of Service Monitoring System for Service Level Agreement Verification", which is supported by Optus through the contract BLO No: 7260. My contributions in this project consisted of three components. Firstly, I compared the performance of different sampling methods for network measurement based on a real traffic trace provided by the WAND group; secondly, I proposed and validated a new adaptive stratified sampling strategy for SLA monitoring; thirdly, I developed a QoS monitoring software, which monitors such QoS parameters as packet delay, packet loss and jitter for SLA monitoring and verification. This software has undergone extensive tests and it has also been evaluated by Optus staff. A discussion for future research is presented at the end of this chapter.

Firstly, I reviewed major publications in the area. A brief introduction to the standard metrics for network measurements was presented, and a detailed discussion of the characteristics of QoS metrics related to the design of the monitoring system in this project (i.e., packet delay, packet loss and jitter) and the challenges in monitoring these metrics was presented. An introduction to active measurements and passive measurements was provided next. An active measurement was employed in the monitoring software to measure RTT delay, one-way loss and jitter.

Secondly, I discussed the most common sampling techniques that are used in sampling-based monitoring systems, such as systematic sampling, random sampling, stratified

random sampling and adaptive sampling. Then a discussion of fundamental limit (e.g., minimum sample size for a given confidence level and an error bound) on the accuracy of sampling techniques was presented.

Thirdly, I compared the performance of different sampling methods by comparing the variance of the sample mean. Several other commonly used metrics such as absolute error of estimated mean and absolute error of estimated variance were also used for comparison. Simulations were performed to validate the theoretical analysis. I used a delay traffic trace as the parent trace, which was generated by the Opnet modeler using a real traffic trace provided by the WAND group. The sample delay trace was selected directly from the parent delay trace instead of obtaining it using active sampling in order to remove the impact of packet size on the delay measurements. Simulation results showed that the count-based systematic sampling produced approximately the same performance as count-based random sampling with only a marginal improvement. The variances of the sample mean of the two timer-based sampling techniques (i.e., timer-based systematic and timer-based Poisson) were expressed in terms of autocorrelation of packet delay of the parent delay trace, and the relationship between the timer-based systematic sampling and the timer-based Poisson sampling was related to the autocorrelation of packet delay and the sampling interval. Simulation results showed that the timer-based systematic sampling achieved better performance than the timer-based Poisson sampling does. The stratified sampling with optimum allocation produces the most accurate estimate, however, it requires *a priori* information (e.g., the standard deviation of packet delay within each stratum of the parent delay trace) which cannot be obtained in a real application.

To address the challenge of implementing stratified sampling with optimum allocation, I proposed an adaptive stratified sampling scheme. The proposed adaptive sampling method does not require *a priori* knowledge of the standard deviations of packet delay. Instead, an LMS algorithm was employed to predict the standard deviation of packet delay from the past observations. A theoretical analysis on the effects of prediction error on the variance of the sample mean was presented. The stratification boundaries were determined based on the autocorrelation of packet delay of the parent delay trace. Simulation results showed that the proposed adaptive strati-

fied sampling method achieved approximately the same performance as the stratified sampling with optimum allocation.

Finally, a detailed introduction to the SLA monitoring software design was presented. The software employs TCP, UDP and ICMP protocols for network measurements and implements several aforementioned sampling schemes. A description of the procedure of the TCP measurement, UDP measurement and ICMP measurement was presented, and the systematic error of the software was calibrated. Finally, an introduction to the software's GUI design, with several test results, was provided.

The software is provided in the attached CD.

7.1 Future Study

In this thesis, I compared the performance of the count-based simple random sampling and the stratified sampling with proportional allocation and the stratified sampling with optimum allocation. But there is no theoretical analysis on the performance comparison between the timer-based sampling (i.e., timer-based systematic sampling and timer-based Poisson sampling) and the stratified sampling. This is a direction for our future research.

As timer-based sampling is easier to implement than the count-based sampling, it is desirable to perform a theoretical analysis to compare the performance of count-based sampling and timer-based sampling. This is another direction for our future study.

Moveover, active measurement methodology is well known to be not scalable because of the intrusive nature of sampling. When the number of clients is large, the large volume of sampling traffic going into a single server may create traffic congestion in the server network and deteriorate the measurement accuracy. This problem can be partly solved by investigating a distributed architecture in which clients send sampling traffic to local servers only and local servers collect the local network performance information. The local information is then sent to a central server, in which the global network performance information is derived. It is our future research di-

rection to investigate and design this architecture for scalable and efficient network performance monitoring.

Bibliography

[1] ITU-T, "Support of ip-based services using ip transfer capabilities," *Tech. Rep. Rec. Y.1241*, 2001.

[2] S. Blake, D. Black, M. Carlson, E. Divies, Z. Wang, and W. Weiss, "An architecture for differentiated services," *IETF RFC 2475*, 1998.

[3] C. Molina-Jimenez, S. Shrivastava, J. Crowcroft, and P. Gevros, "On the monitoring of contractual service level agreements," in *Proceedings of the First International Workshop on Electronic Contracting (WEC'04)*, April, 2004.

[4] K. Claffy, G. Polyzos, and H.-W. Braun, "Application of sampling methodologies to network traffic characterization," *ACM SIGCOMM Computer Communication Review*, vol. 23, no. 4, pp. 194–203, 1993.

[5] K. Claffy, "Internet measurement and data analysis: topology, workload, performance and routing statistics," *NAE'99 workshop*, 1999.

[6] A. Pasztor, "Accurate active measurement in the internet and its application," Doctor of Philosophy thesis, University of Melbourne, 2003.

[7] J. Mahdavi and V. Paxson, "Ippm metrics for measuring connectivity," *IETF RFC 2678*, September, 1999.

[8] G. Almes, S. Kalidindi, and M. Zekauskas, "A one-way delay metric for ippm," *IETF RFC 2679*, 1999.

[9] G. Almes, "A one-way packet loss metric for ippm," *IETF RFC 2680*, 1999.

[10] G. Almes, S. Kalidindi, and M. Zakauskas, "A round-trip delay metric for ippm," *IETF RFC 2681*, 1999.

[11] R. Koodli and R. Ravikanth, "One-way loss pattern sample metrics," *IETF RFC 3357*, 2002.

[12] C. Demichelis and P. Chimento, "Ip packet delay variation metric for ip performance metric (ippm)," *IETF RFC 3393*, 2002.

[13] V. Paxson, G. Almes, J. Mahdavi, and M. Mathis, "Framework for ip performance metrics," *IETF RFC 2330*, May, 1998.

[14] Optus, "Vpn product suite overview," *Tech. Rep*, August 31st 2004.

[15] V. Jacobson, K. Nichols, and K. Poduri, "An expedited forwarding phb," *IETF RFC 2598*, June, 1999.

[16] W. Jiang and H. Schulzrinne, "Modeling of packet loss and delay and their effect on real-time multimedia quality of service quality," in *Proc. Int. Workshop Network and Operating Systems Support for Digital Audio and Video NOSSDAV*, Chapel Hill, NC, 2000.

[17] M. Bjorkman, A. Latour-Henner, U. Hansson, and A. Miah, "Controllability and impact of cell loss process in atm networks," *IEEE Globecom*, pp. 240–248, 1995.

[18] R. Koodli and C. Krishna, "Noticeable loss: a metric for capturing loss pattern for continuous media applications," in *Proceedings of SPIE*, vol. 3529, 1998, pp. 231–239.

[19] H. Sanneck and G. Carle, "A framework model for packet loss metrics based on loss runlengths," in *SPIE/ACM SIGMM Multimedia Computing and Networking Conference 2000 (MMCN 2000)*, San Jose, CA, 2000, pp. 177–187.

[20] M. Yajnik, S. Moon, J. Kurose, and D. Towsley, "Measurement and modelling of the temporal dependence in packet loss," in *IEEE INFOCOM*, vol. 1, 1999, pp. 345–352.

[21] M. Arai, A. Chiba, and K. Iwasaki, "Measurement and modeling of burst packet losses in internet end-to-end communications," in *1999 Pacific Rim International Symposium on Dependable Computing*, 1999, pp. 260–267.

[22] V. Raisanen, "On end-to-end analysis of packet loss," *Computer Communications*, vol. 26, no. 14, pp. 1693–1697, 2003.

[23] J.-C. Bolot, "Characterizing end-to-end packet delay and loss in the internet," *Journal of High-Speed Networks*, vol. 2, no. 3, pp. 305–323, 1993.

[24] M. Hasib, J. Schormans, and J. Pitts, "Probing limitations for packet loss probability measurement on buffered access links," *Electronics Letters*, vol. 40, no. 20, pp. 1315–1316, 2004.

[25] D. Mills, "Simple network time protocol (sntp) version 4 for ipv4, ipv6 and osi," *IETF RFC 2030*, October, 1996.

[26] K. Papagiannaki, S. Moon, C. Fraleigh, P. Thiran, F. Tobagi, and C. Diot, "Analysis of measured single-hop delay from an operational backbone network," in *IEEE INFOCOM 2002*, vol. 2, 2002, pp. 535–544.

[27] K. Papagiannaki, S. Moon, C. Fraleigh, P. Thiran, and C. Diot, "Measurement and analysis of single-hop delay on an ip backbone network," *Selected Areas in Communications, IEEE Journal on*, vol. 21, no. 6, pp. 908–921, 2003.

[28] T. Zseby, "Deployment of sampling methods for sla validation with non-intrusive measurements," in *Proceedings of Passive and Active Measurement Workshop*, 2002.

[29] ——, "Stratification Strategies for Sampling-based Non-intrusive Measurements of One-way Delay," in *The Passive and Active Measurement Workshop*, 2003.

[30] W. G. Cochran, *Sampling Techniques*, 2nd ed. New York: John Wiley and Sons, 1964.

[31] E. Hernandez, M. Chidester, and A. George, "Adaptive Sampling for Network Management," *Journal of Network and Systems Management*, vol. 9, no. 4, 2001.

[32] S. K. Thompson, *Sampling*, 2nd ed. New York: John Wiley and Sons, 2002.

[33] P. D. Amer and L. N. Cassel, "Management of sampled real-time network measurements," in *Local Computer Networks, 1989, Proceedings 14th Conference on*. Mineapolis, MN, USA: IEEE, October, 1989, pp. 62–68.

[34] I. Cozzani and S. Giordano, "Traffic sampling methods for end-to-end QoS evaluation in large heterogeneous networks," *Computer Networks and ISDN Systems*, vol. 30, no. 16-18, pp. 1697–1706, 1998.

[35] Online, "Auckland-VI trace data, http://pma.nlanr.net/Traces/Traces/long/auck/6/."

[36] ——, "http://atm.cs.waikato.ac.nz/wand/wits/auck/6/html/d20010613-060000.html."

[37] J. M. K. Mochalski and S. Donnelly, "Packet delay and loss at the auckland internet access path," in *Proc. Passive Active Measure.*, Workshop, Fort Collins CO(2002), 2002.

[38] P. Calyam and C.-G. Lee, "Characterizing voice and video traffic behavior over the internet," in *International Symposium on Computer and Information Sciences (ISCIS)*, 2005.

[39] P. Brockwell and R. Davis, *Introduction to Time Series and Forecasting*, 2nd ed. New York: Springer, 2002.

[40] R. E. Walpole and R. H. Myers, *Probability and Statistics for Engineers and Scientists*, 2nd ed. New York: Macmillan Publishing Co, 1978.

[41] W. Ma, C. Huang, and J. Yan, "Adaptive sampling for network performance measurement under voice traffic," in *IEEE International Conference on Communications*, vol. 2, 2004, pp. 1129–1134.

[42] B.-Y. Choi, J. Park, and Z.-L. Zhang, "Adaptive random sampling for load change detection," 2001.

[43] E. C.Ifeachor and B. W.Jervis, *Digital Signal Processing-A Practical Approach*, 2nd ed. London: Pearson Education Limited, 2002.

[44] S. Haykin, *Adaptive Filter Theory*, 4th ed. New Jersey: Prentice Hall, 2002.

[45] T. Zseby and S. Zander, "Statistical sla validation based on sampling for highly interactive applications," *Franunhofer FOKUS Report TR-2003-1101*, 2003.

[46] F. Halsall, *Computer Networking and the Internet*, 5th ed. Pearson Education Limited, 2005.

[47] P. Almquist, "Type of service in the internet protocol suite," *IETF RFC 1349*, p. 26, July, 1992.

[48] Online, "IPv6 protocol for Microsoft Windows, http://www.microsoft.com/windowsserver2003/techinfo/overview/ipv6faq.mspx," 2006.

[49] M. Hall, M. Towfiq, G. Arnold, D. Treadwell, and H. Sanders, "Windows Sockets: An Open Interface for Network Programming under Microsoft Windows (Version 1.1)," 1993.

[50] WinSock-Group, "Windows Sockets 2: Application Programming Interface," 1996.

[51] M. Allman, V. Paxson, and W. Stevens, "Tcp congestion control," *IETF RFC 2581*, 1999.

[52] V. Paxson, "Computing TCP's Retransmission Timer," *IETF RFC 2988*, 2000.

[53] S. Deering, "Icmp router discovery messages," *IETF RFC 1256*, September, 1991.

[54] S. Savage, "Sting: a tcp-based network measurement tool," *Proceedings of USENIX Symposium on Internet Technologies and Systems*, 1999.

[55] G. R. Wright and W. R. Stevens, *TCP/IP Illustrated, Volume 2*, 1st ed. Addison Wesley, 1995.

[56] W. R. Stevens, *TCP/IP Illustrated Volume 1*, 1st ed. Addison Wesley, 1994.

[57] "MSDN Library Visual Studio," *Microsoft Corp.*

[58] D. J. Kruglinski, *Inside Visual C++*, 4th ed. Tsinghua University Press, 2002.

Appendix A

Mathematical Derivation

A.1 Derivation of PDF of the sum of m consecutive inter-arrival time slots of the Poisson process

We have obtained the PDF for the 2-th, 3-th, 4-th, 5-th consecutive inter-arrival times of the Poisson process in Chapter 4.2. They are:

$$p_2(t) = \lambda^2 t e^{-\lambda t}, \tag{A.1}$$

$$p_3(t) = \frac{1}{2}\lambda^3 t^2 e^{-\lambda t}, \tag{A.2}$$

$$p_4(t) = \frac{1}{6}\lambda^4 t^3 e^{-\lambda t}, \tag{A.3}$$

$$p_5(t) = \frac{1}{24}\lambda^5 t^4 e^{-\lambda t}. \tag{A.4}$$

From Equation A.1, A.2, A.3 and A.4, we can find that their expressions have the same form. Therefore, by mathematical induction, we suppose that when $m \leq k$, $p_m(t) = \frac{1}{(m-1)!}\lambda^m t^{m-1} e^{-\lambda t}$. Then when $k = k+1$, let $F_{k+1}(t)$ denote the distribution function of $(k+1)$-th consecutive inter-arrival times:

$$F_{k+1}(t) = P\{T \leq t\} = P\{X + Y \leq t\}. \tag{A.5}$$

where the PDF of X is $p_k(t)$ and the PDF of Y is $p_1(t)$. Then,

$$F_{k+1}(t) \quad = \quad \int_0^t \int_0^{t-x} p_k(x) p_1(y) dx dy, \tag{A.6}$$

$$= \int_0^t \frac{1}{(k-1)!}\lambda^k x^{k-1} e^{-\lambda x} dx \int_0^{t-x} \lambda e^{-\lambda y} dy, \quad \text{(A.7)}$$

$$= \int_0^t \frac{1}{(k-1)!}\lambda^k x^{k-1} e^{-\lambda x} dx (1 - e^{-\lambda(t-x)}), \quad \text{(A.8)}$$

$$= \int_0^t \frac{1}{(k-1)!}\lambda^k x^{k-1} e^{-\lambda x} dx - \int_0^t \frac{1}{(k-1)!}\lambda^k x^{k-1} e^{-\lambda t} dx, \quad \text{(A.9)}$$

$$= \int_0^t \frac{1}{(k-1)!}\lambda^k x^{k-1} e^{-\lambda x} dx - \frac{\lambda^k}{k!} t^k e^{-\lambda t}. \quad \text{(A.10)}$$

Then,

$$p_{k+1}(t) = F'_{k+1}(t), \quad \text{(A.11)}$$

$$= \frac{1}{(k-1)!}\lambda^k t^{k-1} e^{-\lambda t} - \frac{\lambda^k}{k!} k t^{k-1} e^{-\lambda t} + \frac{\lambda^k}{k!} t^k \lambda e^{-\lambda t}, \quad \text{(A.12)}$$

$$= \frac{\lambda^{k+1}}{k!} t^k e^{-\lambda t}. \quad \text{(A.13)}$$

Hence, when $k = k + 1$, the hypothesis is also valid.

A.2 Derivation of comparison between the variance of the systematic sample mean and the variance of the Poisson sample mean

We have obtained the variance of the systematic sample mean and the variance of the Poisson sample mean in Chapter 4.2. They are:

$$Var_s(\bar{y}) = \frac{\sigma^2}{n}(1 + 2\sum_{h=1}^{n-1}\frac{n-h}{n}ae^{-bhT_0}), \tag{A.14}$$

$$Var_p(\bar{y}) = \frac{\sigma^2}{n}(1 + 2\sum_{h=1}^{n-1}\frac{n-h}{n}\frac{a\lambda^h}{(b+\lambda)^h}), \tag{A.15}$$

Because $1/\lambda = T_0$, the Equation A.15 can be simplified as:

$$Var_p(\bar{y}) = \frac{\sigma^2}{n}(1 + 2\sum_{h=1}^{n-1}\frac{n-h}{n}\frac{a}{(1+bT_0)^h}). \tag{A.16}$$

If

$$\frac{n-h}{n}ae^{-bhT_0} < \frac{n-h}{n}\frac{a}{(1+bT_0)^h}, \tag{A.17}$$

$$\Rightarrow -bhT_0 < \ln\frac{1}{(1+bT_0)^h}, \tag{A.18}$$

$$\Rightarrow -bhT_0 < -h\ln(1+bT_0), \tag{A.19}$$

$$\Rightarrow bT_0 > \ln(1+bT_0). \tag{A.20}$$

Vice versa. Therefore

$$\frac{n-h}{n}ae^{-bhT_0} < \frac{n-h}{n}\frac{a}{(1+bT_0)^h} \Leftrightarrow bT_0 > ln(1+bT_0). \tag{A.21}$$

In the same way, we can obtain:

$$\frac{n-h}{n}ae^{-bhT_0} > \frac{n-h}{n}\frac{a}{(1+bT_0)^h} \Leftrightarrow bT_0 < ln(1+bT_0), \tag{A.22}$$

$$\frac{n-h}{n}ae^{-bhT_0} = \frac{n-h}{n}\frac{a}{(1+bT_0)^h} \Leftrightarrow bT_0 = ln(1+bT_0). \tag{A.23}$$

A.2.1 Sufficient condition

If $bT_0 < ln(1 + bT_0)$, then, for all $h = 1, 2, ..., n - 1$,

$$\frac{n-h}{n} ae^{-bhT_0} > \frac{n-h}{n} \frac{a}{(1+bT_0)^h}, \tag{A.24}$$

$$\Rightarrow \sum_{h=1}^{n-1} \frac{n-h}{n} ae^{-bhT_0} > \sum_{h=1}^{n-1} \frac{n-h}{n} \frac{a}{(1+bT_0)^h}, \tag{A.25}$$

$$\Rightarrow \frac{\sigma^2}{n}(1 + 2\sum_{h=1}^{n-1} \frac{n-h}{n} ae^{-bhT_0}) > \frac{\sigma^2}{n}(1 + 2\sum_{h=1}^{n-1} \frac{n-h}{n} \frac{a}{(1+bT_0)^h}), \tag{A.26}$$

$$\Rightarrow Var_s(\bar{y}) > Var_p(\bar{y}). \tag{A.27}$$

A.2.2 Necessary condition

If $Var_s(\bar{y}) > Var_p(\bar{y})$, then

$$\frac{\sigma^2}{n}(1 + 2\sum_{h=1}^{n-1} \frac{n-h}{n} ae^{-bhT_0}) > \frac{\sigma^2}{n}(1 + 2\sum_{h=1}^{n-1} \frac{n-h}{n} \frac{a}{(1+bT_0)^h}). \tag{A.28}$$

We can obtain:

$$\Rightarrow \sum_{h=1}^{n-1} \frac{n-h}{n} ae^{-bhT_0} > \sum_{h=1}^{n-1} \frac{n-h}{n} \frac{a}{(1+bT_0)^h}. \tag{A.29}$$

Assuming there exists $i\epsilon[1, n-1]$ that satisfies $\frac{n-i}{n} ae^{-biT_0} < \frac{n-i}{n} \frac{a}{(1+bT_0)^i}$, then we can obtain $bT_0 > ln(1 + bT_0)$:

$$\Rightarrow Var_s(\bar{y}) < Var_p(\bar{y}). \tag{A.30}$$

This is incompatible with the original condition $Var_s(\bar{y}) > Var_p(\bar{y})$, so there is no such i existing.

Assuming there exists $i\epsilon[1, n-1]$ that satisfies $\frac{n-i}{n} ae^{-biT_0} = \frac{n-i}{n} \frac{a}{(1+bT_0)^i}$, then we can obtain $bT_0 = ln(1 + bT_0)$:

$$\Rightarrow Var_s(\bar{y}) = Var_p(\bar{y}). \tag{A.31}$$

This is incompatible with the original condition $Var_s(\bar{y}) > Var_p(\bar{y})$, so there is no such i existing. So for all $h = 1, 2, ...n - 1$, $\frac{n-h}{n} ae^{-bhT_0} < \frac{n-h}{n} \frac{a}{(1+bT_0)^h}$, $\Rightarrow bT_0 < ln(1 + bT_0)$.

We therefore proved that:

$$bT_0 < ln(1 + bT_0) \Leftrightarrow Var_s(\bar{y}) > Var_p(\bar{y}). \tag{A.32}$$

In the same way, we can prove that:

$$bT_0 > ln(1 + bT_0) \Leftrightarrow Var_s(\bar{y}) < Var_p(\bar{y}), \tag{A.33}$$

$$bT_0 = ln(1 + bT_0) \Leftrightarrow Var_s(\bar{y}) = Var_p(\bar{y}). \tag{A.34}$$

A.3 Derivation of the transformation of σ^2 in terms of the autocorrelation function

For stratified sampling, it can be shown that the variance of the parent population (i.e. the true variance) is related to values in each stratum by:

$$\begin{aligned}
\sigma^2 &= \frac{1}{N-1}\sum_{i=1}^{N}(y_i-\mu)^2, && \text{(A.35)}\\
&= \frac{1}{N-1}\sum_{l=1}^{L}\sum_{i=1}^{N_l}(y_{li}-\mu)^2, && \text{(A.36)}\\
&= \frac{1}{N-1}\sum_{l=1}^{L}\sum_{i=1}^{N_l}(y_{li}-\mu_l+\mu_l-\mu)^2, && \text{(A.37)}\\
&= \frac{1}{N-1}\sum_{l=1}^{L}\sum_{i=1}^{N_l}[(y_{li}-\mu_l)+(\mu_l-\mu)]^2, && \text{(A.38)}\\
&= \frac{1}{N-1}\sum_{l=1}^{L}\sum_{i=1}^{N_l}(y_{li}-\mu_l)^2+\frac{1}{N-1}\sum_{l=1}^{L}\sum_{i=1}^{N_l}2(y_{li}-\mu_l)(\mu_l-\mu), && \text{(A.39)}\\
&\quad +\frac{1}{N-1}\sum_{l=1}^{L}\sum_{i=1}^{N_l}(\mu_l-\mu)^2. && \text{(A.40)}
\end{aligned}$$

Consider the second term in Equation A.39. It can be shown that:

$$\begin{aligned}
\sum_{i=1}^{N_l}2(y_{li}-\mu_l)(\mu_l-\mu) &= 2(\mu_l-\mu)\sum_{i=1}^{N_l}(y_{li}-\mu_l), && \text{(A.41)}\\
&= 2(\mu_l-\mu)(\sum_{i=1}^{N_l}y_{li}-N_l\mu_l), && \text{(A.42)}\\
&= 2(\mu_l-\mu)(N_l\mu_l-N_l\mu_l), && \text{(A.43)}\\
&= 0. && \text{(A.44)}
\end{aligned}$$

From the earlier derivation, the second term in Equation A.39 should be zero, i.e.,

$$\frac{1}{N-1}\sum_{l=1}^{L}\sum_{i=1}^{N_l}2(y_{li}-\mu_l)(\mu_l-\mu)=0. \quad \text{(A.45)}$$

Therefore the earlier equation on σ^2 can be simplified as:

$$\sigma^2 = \frac{1}{N-1}\sum_{l=1}^{L}\sum_{i=1}^{N_l}(y_{li}-\mu_l)^2+\frac{1}{N-1}\sum_{l=1}^{L}\sum_{i=1}^{N_l}(\mu_l-\mu)^2, \quad \text{(A.46)}$$

$$= \frac{1}{N-1}\sum_{l=1}^{L}(N_l-1)\frac{1}{N_l-1}\sum_{i=1}^{N_l}(y_{li}-\mu_l)^2 + \frac{1}{N-1}\sum_{l=1}^{L}N_l(\mu_l-\mu)^2, \quad \text{(A.47)}$$

$$= \frac{1}{N-1}\sum_{l=1}^{L}(N_l-1)\sigma_l^2 + \frac{1}{N-1}\sum_{l=1}^{L}N_l(\mu_l-\mu)^2. \quad \text{(A.48)}$$

A.4 Derivation of the difference of the variance of the sample mean between optimum allocation and proportional allocation

$$
\begin{aligned}
& Var_{prop}(\bar{y}) - Var_{opt}(\bar{y}) && \text{(A.49)}\\
= \; & \frac{1}{n}(1-\frac{n}{N})\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l^2 - [\frac{1}{n}(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)^2 - \frac{1}{N}\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l^2], && \text{(A.50)}\\
= \; & \frac{1}{n}\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l^2 - \frac{1}{n}(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)^2, && \text{(A.51)}\\
= \; & \frac{1}{nN}[\sum_{l=1}^{L}N_l\sigma_l^2 - \frac{1}{N}(\sum_{l=1}^{L}N_l\sigma_l)(\sum_{l=1}^{L}N_l\sigma_l)], && \text{(A.52)}\\
= \; & \frac{1}{nN}[\sum_{l=1}^{L}N_l\sigma_l^2 - (\sum_{l=1}^{L}N_l\sigma_l)(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)], && \text{(A.53)}\\
= \; & \frac{1}{nN}[\sum_{l=1}^{L}N_l\sigma_l^2 - (2\sum_{l=1}^{L}N_l\sigma_l - \sum_{l=1}^{L}N_l\sigma_l)(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)], && \text{(A.54)}\\
= \; & \frac{1}{nN}[\sum_{l=1}^{L}N_l\sigma_l^2 - 2\sum_{l=1}^{L}N_l\sigma_l(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l) + \sum_{l=1}^{L}N_l\sigma_l(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)], && \text{(A.55)}\\
= \; & \frac{1}{nN}[\sum_{l=1}^{L}N_l\sigma_l^2 - 2\sum_{l=1}^{L}N_l\sigma_l(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l) + \frac{\sum_{l=1}^{L}N_l}{N}\sum_{l=1}^{L}N_l\sigma_l(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)], && \text{(A.56)}\\
= \; & \frac{1}{nN}[\sum_{l=1}^{L}N_l\sigma_l^2 - 2\sum_{l=1}^{L}N_l\sigma_l(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l) + \sum_{l=1}^{L}N_l(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)], && \text{(A.57)}\\
= \; & \frac{1}{nN}\sum_{l=1}^{L}N_l[\sigma_l^2 - 2\sigma_l(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l) + (\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)^2], && \text{(A.58)}\\
= \; & \frac{1}{nN}\sum_{l=1}^{L}N_l(\sigma_l - \bar{\sigma_l})^2. && \text{(A.59)}
\end{aligned}
$$

A.5 Derivation of the relative error between $Var_{opt}(\bar{y})$ and $Var_{act}(\bar{y})$

In Chapter 5.3.3, we obtained that:

$$Var_{act}(\bar{y}) = \sum_{l=1}^{L}(\frac{N_l}{N})^2\frac{\sigma_l^2}{\hat{n}_l} - \sum_{l=1}^{L}(\frac{N_l}{N})^2\frac{\sigma_l^2}{N_l}, \tag{A.60}$$

$$Var_{opt}(\bar{y}) = \frac{1}{n}(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)^2 - \frac{1}{N}\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l^2. \tag{A.61}$$

Then,

$$\begin{aligned} Var_{act}(\bar{y}) - Var_{opt} &= \sum_{l=1}^{L}(\frac{N_l}{N})^2\frac{\sigma_l^2}{\hat{n}_l} - \frac{1}{n}(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)^2, & \text{(A.62)}\\ &= \sum_{l=1}^{L}(\frac{N_l}{N})^2\frac{\sigma_l^2}{n_l^2}\frac{n_l^2}{\hat{n}_l} - \frac{1}{n}(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)^2. & \text{(A.63)} \end{aligned}$$

Substituting n_l^2 in the denominator of the first term of the Equation A.63 by Equation 4.66:

$$\begin{aligned} Var_{act}(\bar{y}) - Var_{opt} &= \sum_{l=1}^{L}\frac{N_l^2}{N^2}\frac{\sigma_l^2(\sum_{k=1}^{L}N_k\sigma_k)^2}{N_l^2\sigma_l^2n^2}\frac{n_l^2}{\hat{n}_l} - \frac{1}{n}(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)^2, & \text{(A.64)}\\ &= \frac{1}{n^2}(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)^2\sum_{l=1}^{L}\frac{n_l^2}{\hat{n}_l} - \frac{1}{n}(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)^2, & \text{(A.65)}\\ &= \frac{1}{n^2}(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)^2(\sum_{l=1}^{L}\frac{n_l^2}{\hat{n}_l} - n). & \text{(A.66)} \end{aligned}$$

Based on the theoretical analysis in Chapter 5.3.1, we have $E(\sum_{l=1}^{L}\hat{n}_l) = E(\sum_{l=1}^{L}n_l) = n$.

Therefore, Equation A.66 can be further simplified as:

$$\begin{aligned} Var_{act}(\bar{y}) - Var_{opt}(\bar{y}) &= \frac{1}{n^2}(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)^2\sum_{l=1}^{L}(\frac{n_l^2}{\hat{n}_l} - 2n_l + \hat{n}_l), & \text{(A.67)}\\ &= \frac{1}{n^2}(\sum_{l=1}^{L}\frac{N_l}{N}\sigma_l)^2\sum_{l=1}^{L}\frac{(\hat{n}_l - n_l)^2}{\hat{n}_l}. & \text{(A.68)} \end{aligned}$$

Ignoring the fpc, the relative error of actual variance of the sample mean is:

$$\frac{Var_{act}(\bar{y}) - Var_{opt}(\bar{y})}{Var_{OPT}(\bar{y})} = \frac{1}{n}\sum_{l=1}^{L}\frac{(\hat{n}_l - n_l)^2}{\hat{n}_l}. \tag{A.69}$$