Measurements, Analysis, and Modeling for Network Throughput Prediction on the Internet

January, 2013

Doctor of Engineering

Chunghan Lee

Toyohashi University of Technology

Measurements, Analysis, and Modeling for Network Throughput Prediction on the Internet

Abstract

Data volume transferred on the Internet is growing explosively along with the advancement of computer technology and the popularization of Internet applications. To take account of the data volume, introducing alternative systems or redesigning the whole Internet might be adequate but cannot be achieved promptly. Introducing bandwidth reservation mechanism is one of possible directions of the adequate evolution. However, dedicated network infrastructures would be necessary for implementing such a kind of mechanisms. On the other hand, a method that efficiently utilizes the existing Internet is promising. Network throughput prediction on the Internet, which is another approach for the efficient utilization of the Internet, does not require hardware for the infrastructures, and it can help to enhance site selection on multiple sites and grid applications by reducing data transfer time.

Although many solutions have been proposed for predicting the network throughput, they suffer the following obstacles. First, it is hard to be modeled mathematically because distribution of traffic fluctuation is unclear. Second, changes in data are very large because the scale and bandwidth of network are rapidly increased each year. Third, there is noise occurred by abrupt changes in network state.

In this thesis, throughput prediction methods and its application to efficient utilization of the Internet are explored.

As a first step, we introduce and discuss research issues of Internet traffic characteristics when virtualization technology is used on the Internet. We gather throughput measurements on the Internet and appropriate prediction parameters are selected through statistical analysis before the actual prediction. In the throughput measurements, we clarify extra effects caused by the virtualization. Although network state is stable, the measurement results are fluctuated by the effects. A previous prediction parameter is inappropriate for the precise prediction, and noise and non-linear characteristics are found.

Next, we propose a throughput prediction method with improved prediction results. Machine learning techniques, that find patterns or characteristic features, are applied

on the throughput measurements to automatically determine an appropriate regression curve and to deal with prediction models represented as noise and non-linear characteristics. The prediction results of our method compared to those of the previous prediction method are more accurate on the same condition.

Finally, we investigate the contributions of prediction results through grid simulation. Our prediction method and the previous prediction method were adopted by a grid scheduler and their performance was quantitatively compared to that of the scheduler without any throughput predictions. Through the simulation, the prediction results will not always contribute to reduce processing time on given tasks. Most results reduce overall processing time using our prediction method.

Contents

Abstract **ii**

List of Figures **viii**

List of Tables **xi**

1 Introduction **1**
1.1 Background 1
1.2 Anomaly 2
1.3 Research Problems and Questions 3
1.4 Characteristics of Thesis 6
1.4.1 Networking 6
1.4.2 Machine Learning 6
1.5 Contributions of Thesis 7
1.6 Thesis Organization 8

2 Traffic Characteristics on the Internet **10**
2.1 Overview 10
2.2 Precise Network Measurement on Virtualized Environments 11
2.3 Packet Spacing 12
2.3.1 Concept of Packet Spacing 12
2.3.2 Necessity for Estimating Oversize Packet Spacing on Userland . 13
2.4 Measurement Methodology 13
2.4.1 PlanetLab as a Virtualized Network Testbed 13
2.4.2 Node Selection 14
2.4.3 Connection pair 16
2.4.4 Throughput Measurement using Connection Pair 16
2.5 Throughput Measurement Results 18
2.5.1 Theoretical Achievable Throughput 18
2.5.2 The Statistics of Measurement Results 18
2.6 Packet-level Analysis 19

2.6.1 Round-trip Time (RTT) 20
2.6.2 Packet Loss Rate 21
2.6.3 Cause of Network Throughput Drop 22
2.6.4 Advertised Window 24
2.6.5 Anomalous Case 26
2.7 Naive Method for Estimating Anomalous Case 30
2.7.1 CPU Utilization 30
2.7.2 CPU Availability 32
2.8 Traffic Characteristics on Cloud Service 33
2.8.1 Amazon EC2 33
2.8.2 Performance Analysis of Amazon EC2 33
2.8.3 Throughput Measurement 34
2.8.4 Resource Monitoring Method 35
2.8.5 Throughput Measurement Results 36
2.8.6 Analysis of Network Metrics 37
2.8.6.1 RTT and Packet Loss 37
2.8.6.2 Packet Spacing 39
2.8.7 CPU Utilization 43
2.9 Conclusion 44

3 Estimation of Traffic Anomaly on the Internet 45
3.1 Overview 45
3.2 Problem of Naive Method 45
3.3 Monitoring Systems 46
3.4 Throughput Measurement with Resource State 46
3.4.1 Node Characteristics 46
3.4.2 Throughput Measurement Results 47
3.4.3 Cause of Throughput Instability 48
3.4.4 Features of Resource State 49
3.5 A Method for Estimating Anomalous Case using PCA 51
3.5.1 Principal Component Analysis (PCA) 52
3.5.2 Analysis Result using PCA 53
3.5.3 Boundaries for Estimating Anomalous Case 55
3.6 Evaluation Results using PCA 56
3.6.1 Input Data Set 56
3.6.2 Evaluation Results 58
3.7 Conclusion 59

4 Parameter Selection using Statistical Approach 60
4.1 Overview 60
4.2 Proposed Approach for Parameter Selection 61
4.3 Evaluation Results for the Selection of Probe Size 61
4.4 Packet-level Analysis 62
4.4.1 The Cause of Throughput Fluctuations 63

4.4.2 Appropriate Probe Size for Prediction Model 66
4.5 Actual Throughput using Various Probe Sizes 66
4.6 Conclusion 68

5 Analytical Modeling for Network Throughput Prediction 69
5.1 Overview 69
5.2 Related Work 69
5.2.1 Prediction of Network Traffic 70
5.2.2 Throughput Prediction Methods 70
5.2.3 SVM and SVR for Network Research 72
5.3 Data Sets for Throughput Prediction 72
5.3.1 Training Data Sets 73
5.3.2 Input Data Sets 74
5.4 Previous Prediction Method (CDF-based Predictor) 75
5.4.1 Building CDF-based Predictor 75
5.4.2 Prediction Results 77
5.5 Proposed Prediction Method (SVR-based Predictor) 77
5.5.1 SVR Overview 77
5.5.2 Building SVR-based Predictor 79
5.6 Comparison of Prediction Results 81
5.7 Issue of Probe Size 86
5.7.1 SVR-based Predictor with 512-KB Probe 86
5.7.2 SVR-based Predictor with 64-KB Probe 88
5.8 Issue of Predictability using Non-parametric Regression Techniques . . 91
5.8.1 Non-parametric Regression Techniques 91
5.8.1.1 GAM Overview 92
5.8.1.2 MARS Overview 93
5.8.2 Prediction Results 94
5.9 Conclusion 96

6 Performance Implications by Predicting Throughput 97
6.1 Overview 97
6.2 Motivation 98
6.3 Grid Services and Schedulers 98
6.4 Measurement Methodology 100
6.4.1 Training Data Sets 100
6.4.2 Input Data Sets 102
6.5 Throughput Prediction Methods 103
6.5.1 Previous Throughput Predictor (CDF-based Predictor) 103
6.5.2 Proposed Throughput Predictor (SVR-based Predictor) 103
6.5.3 Evaluation of Prediction Results 104
6.6 Simulation Method 106
6.6.1 Model 106
6.6.2 Task Insert Periods 107

6.6.3 Meta-schedulers . . . 108
6.6.4 Scenario . . . 109
6.7 Evaluation of Simulation Results . . . 110
6.7.1 Scheduling Performance Rate (SPR) . . . 110
6.7.2 Simulation Results . . . 111
6.7.3 Cause of Decreased Scheduling Performance . . . 113
6.7.4 SPR of Entire Result Set . . . 115
6.7.5 Simulation Results without Site S-3 . . . 116
6.8 Implications . . . 118
6.9 Conclusion . . . 121

7 Conclusion and Future Directions **122**
7.1 Overview . . . 122
7.2 Summary . . . 122
7.3 Contributions . . . 125
7.4 Future Directions . . . 128
7.4.1 Issue of Improving Predictability . . . 128
7.4.2 Issue of Design and Implements . . . 129
7.5 Closing Statement . . . 130

Bibliography **131**

Acknowledgements **145**

Lists of publications **146**

List of Figures

1.1 Actual throughput at cases A and B. 3
1.2 Case of connection state on virtualized environment. 4
1.3 Composition of bandwidth on end nodes. 5
1.4 Problem of network throughput prediction. 5
1.5 Thesis Organization. 8

2.1 Concept of packet spacing. 12
2.2 PlanetLab architecture. 14
2.3 RTT using ping at pair (κ, λ). 15
2.4 Connection pair measurement method with resource monitoring. 17
2.5 Throughput measurement results (probe: 64 KB, data : 32 MB). 19
2.6 CDF of RTT of data transfer at all node pairs (32MB). 20
2.7 RTT, Estimated RTT, and Timeout of data transfer at pair (κ, λ). 21
2.8 CDF of packet spacing for connection pair at all pairs. 23
2.9 Mean packet spacing for connection pair at all senders (probe: 64 KB, data : 32 MB). 24
2.10 Packet spacing and advertised window. 25
2.11 CDF of RTT and packet spacing at λ (32 MB). 27
2.12 CDF of RTT at anomalous case (32 MB). 28
2.13 CDF of packet spacing at anomalous case (32 MB). 29
2.14 CPU utilization at all sender nodes. 31
2.15 Minimum, mean, and maximum CPU utilization at all sender nodes. . . 31
2.16 CPU availability at all sender nodes. 32
2.17 Throughput measurement method on Amazon EC2. 35
2.18 Connection pair throughput at indices *i* and *vi*. 36
2.19 RTT of connection pair at index *ix*. 38
2.20 CDF of mean packet spacing at all indices on Amazon EC2, PlanetLab, and local environment. 40
2.21 Packet spacing of outgoing transfers at index *ix*. 41
2.22 Packet spacing and advertised window of outgoing transfers at *ix*. . . . 42
2.23 CPU utilization on Amazon EC2 (indices *i* and *vi*). 43

3.1 Actual throughput for connection pair at all pairs. 48
3.2 Mean packet spacing and actual throughput at all sender nodes. 50
3.3 CDF of packet spacing at node δ. 51
3.4 Minimum, mean, and maximum CpuR at all sender nodes. 52

3.5 First and second component scores. 54
3.6 Component scores with boundaries. 56
3.7 Mean packet spacing and actual throughput at nodes σ and ω. 57
3.8 Component scores with boundaries and input data set. 58

4.1 Measurement methodology. 62
4.2 Daily change in transfer time at index *iii*. 63
4.3 CDF of packet spacing for connection pair at cases A and B. 64
4.4 Mean packet spacing and throughput at index *iii*. 65
4.5 Actual throughput for various connection pairs. 67

5.1 Actual throughput for connection pair at node pair (η, θ). 74
5.2 Actual throughput of input data set at node pair (η, θ). 75
5.3 CDF of connection pair at node pair (η, θ). 76
5.4 Prediction results of CDF-based predictor at node pair (κ, λ). 77
5.5 SVR-based predictor at node pair (η, θ). 80
5.6 Fraction of RPE within 10% or less for input data sets. 82
5.7 Prediction results at node pair (η, θ). 83
5.8 Fraction of RPE within 10% or less for additional input data sets. 83
5.9 Prediction results at node pair (κ, λ). 84
5.10 Prediction results with degree 3 and 5 at node pair (κ, λ). 85
5.11 Actual throughput of training data sets at node pair (η, θ). 86
5.12 SVR-based predictor with 512-KB probe at node pair (η, θ). 87
5.13 Fraction of RPE within 10% or less with 512-KB probes. 88
5.14 Actual throughput of training data sets at node pair (γ, δ). 89
5.15 Actual throughput of training data sets at node pair (η, θ). 89
5.16 Prediction results at node pair (η, θ). 90
5.17 Fraction of RPE within 10% or less with 64-KB probes. 91
5.18 Prediction results using SVR, GAM, and MARS at node pair (η, θ). . . 94
5.19 Prediction results using SVR, GAM, and MARS at node pair (κ, λ). . . 95
5.20 A part of prediction results using SVR, GAM, and MARS at node pair (κ, λ). 95

6.1 Actual throughput of connection pair at site S-4. 101
6.2 Time-series data of connection pair at site S-4. 102
6.3 CDF-based predictor at site S-4. 103
6.4 SVR-based predictor at site S-4. 104
6.5 Fraction of RPE within 10% or less. 105
6.6 Prediction results at S-4. 105
6.7 Simulation model. 107
6.8 SPR of simulation results in idle state. 112
6.9 SPR of simulation results in busy state. 112
6.10 Gantt chart of idle state for task set (II). 114
6.11 Gantt chart of busy state for task set (II). 114
6.12 CDF of SPR for entire result set. 116

6.13 SPR of simulation results in idle state without site S-3. 117
6.14 SPR of simulation results in busy state without site S-3. 118
6.15 Scatter plots of correlation coefficient at NT(SVR). 120
6.16 Scatter plots of correlation coefficient at NT(CDF). 120

List of Tables

1.1 Mean throughput and network metrics. 3

2.1 Node characteristics on virtualized network testbed. 15
2.2 Probe and data sizes for connection pair. 17
2.3 Ideal, minimum, mean, and maximum throughput. 19
2.4 Mean RTT of connection pair. 20
2.5 Mean packet loss rate of connection pair. 21
2.6 Probe and data size combinations for connection pairs. 35
2.7 Mean values of well-known network metrics. 37

3.1 Node characteristics on virtualized testbed. 47
3.2 Mean RTT and packet loss rate of connection pair. 48
3.3 Mean usage of resource state at all sender nodes. 51
3.4 Eigenvalues and cumulative percentage of total variance. 53
3.5 Component loadings. 53
3.6 Node characteristics of input data set on virtualized testbed. 58
3.7 Mean usage of resource state at input data set. 58

4.1 Spearman's rank correlation coefficient for connection pairs. 62
4.2 Evaluation results without anomalous cases. 66

5.1 Geographic location and mean RTT at node pairs. 73
5.2 Statistics of training data sets. (NT is network throughput.) 74
5.3 Composition of input data sets. (NT is network throughput.) 75
5.4 Root-Mean-Square error of input data sets. 82
5.5 Root-Mean-Square error of additional input data sets. 83
5.6 Fraction of RPE of degrees for input data sets. (Deg. is degree). 85
5.7 Root-Mean-Square error with 256-KB and 512-KB probes. 88
5.8 Root-Mean-Square error with 256-KB and 64-KB probes. 90

6.1 Statistics of learning data sets. 101
6.2 Characteristics of input data sets. 102
6.3 Number of tasks at task sets for simulation. 107
6.4 Expected values of SPR for entire result set. 115
6.5 SPR statistics for entire result set. 116
6.6 Correlation coefficient between SPR and statistics of RPE. 119

Chapter 1

Introduction

1.1 Background

The Internet is an essential infrastructure, and many researchers and developers have been devoted for the safe and efficient use of the Internet. Therefore, Internet applications [1][2] and cloud services [3][4] have been widely used to save user's data and to share the data to many people. Recently, the volume of data on the Internet is rapidly increased through the increasing the Internet applications and the development of network equipments. In order to take account such volume, bandwidth reservation would be a reasonable way, but it can be promptly impossible to apply the system to the Internet. Under this situation, a method that efficiently uses the Internet is an alternative way. For an efficient use of the Internet, the applications have considered network characteristics. With the characteristics, the applications can control transfer timing or a volume of data transfer to avoid network congestion. Moreover, we can select a fast server with the characteristics to reduce data transfer time.

In grid [5], network is used for moving data to distributed resources. Thus, overall processing time is affected by network resource. If we can use network resource efficiently, we can reduce the overall processing time. We here introduce how to use network resource on grid applications. To share huge amounts of data on the grid applications to distributed resources, well-known network metrics, such as Round-trip Time (RTT), packet loss, and so on or reserving communication resources are provided to grid schedulers [6][7]. However, it would be hard to provide the metrics on current Internet environments because Internet service providers (ISPs) do not disclose the metrics on their network environments. Next, dedicated communication infrastructures are required for such a kind of reservation mechanisms. A construction of infrastructures on the Internet would be a possible solution, but it is costly and takes a lot of time for the construction.

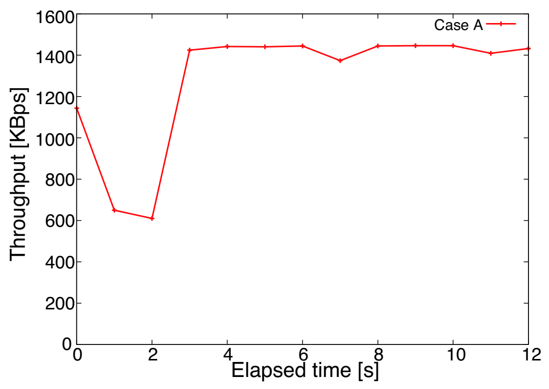
Network throughput prediction on the Internet is a sensible solution for the efficient use. It does not require hardware for the infrastructures. Moreover, it can be used to enhance grid task scheduling, path selection on multiple paths, and the efficiency of data transfer. The predicted throughput can be provided as a criterion to schedule grid tasks and the path that has the shortest transfer time can be selected by using the predicted throughput.

In this thesis, we report on a throughput prediction method on the Internet with improved prediction accuracy. We focus on measurements, analysis, and modeling for precise prediction results. We also discuss many issues of traffic characteristics on the current Internet, describe the prediction method, and clarify the effect of prediction results to the grid applications.

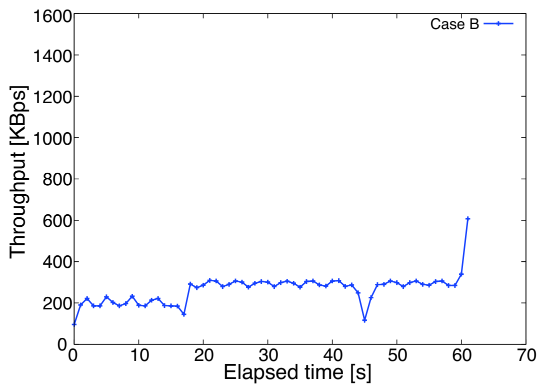
1.2 Anomaly

In general, *anomaly* is something that deviates from what is normal, expected, and predictable. The anomaly has been widely used in many different areas. For example, the anomaly in stock market is a distortion in prices. Thus, the meaning of anomaly would be different depending on the definition. To prevent confusion with the meaning, the definition of anomaly in this thesis is as follows: a situation determined by some measurement result. If we observe that network bandwidth is significantly slower than that of our expectation from the network metrics or past measurement results, we judge that there is the anomaly. In order to determine the anomaly, preconditions are analysis results in packet-level. In detail, throughput instability occurred by virtualization technology is the representative anomaly. The precondition is stable network state where there are no significant changes in well-known network metrics, such as RTT, packet loss rate, and advertised window.

For a better understanding of the anomaly, we prepare two throughput measurement results (cases A and B) on the Internet. The same sender and receiver nodes are used for the measurements, and the data size transmitted to the receiver is 16 MB (megabytes). The actual throughput is shown in Figure 1.1. Despite of the same data size, the actual throughput at case B is decreased in comparison with that at case A. Normally, we can assume a cause of throughput instability at case B as network congestion. Thus, the cause of throughput instability is predicable. To clarify the cause, we analyze the results in packet-level. The mean throughput and network metrics are shown in Table 1.1. Although the actual throughput at case B is decreased, there are no significant differences in other network metrics. Case B is a typical case we call it *anomaly*. Again, we focus on the measurement result for the definition of anomaly. We will introduce and discuss the anomaly deeply in Chapter 2.



(a) Actual throughput at case A



(b) Actual throughput at case B

FIGURE 1.1: Actual throughput at cases A and B.

TABLE 1.1: Mean throughput and network metrics.

	Data size [MB]	**Mean throughput [KBps]**	**Mean RTT [s]**	**Packet loss rate [%]**
Case A	16	1266.9	0.0442	0.196
Case B	16	259.5	0.0451	0.065

1.3 Research Problems and Questions

In this section, we introduce problems that this research deals with. We first describe the situation on the current Internet environment and the reason why Internet traffic characteristics are considered. Second, we explain a composition of bandwidth on end nodes to understand bandwidth. Third, we introduce what is network throughput. Next, the prediction of network throughput in this research is described. Finally, we show research questions for the accurate throughput prediction.

A virtualization technology [8][9] is a promising technology that provides software environments in the form of virtual machines (VMs) [10][11][12], and many fields of network have been widely used. Thus, network measurement is different from non-virtualized environments. Because the VMs on a physical node are provided as platforms for user, lower layers, such as TCP/IP, are hidden and virtualized. Under this situation, extra effects may be induced by the virtualization. Then, real resource state on a physical node is hidden and it would be hard to track user behaviors with the virtual resource state. For example, a case of connection state on virtualized environment is shown in Figure 1.2. Although VM1 and VM3 on site A are connected to VM1 and VM2 on site B, each VM cannot track behaviors of other VMs on the same physical node. In order to create a precise throughput prediction, we should consider above situation. Moreover, we should also investigate the current behavior of Internet traffic because the bandwidth and scale of networks are rapidly increased.

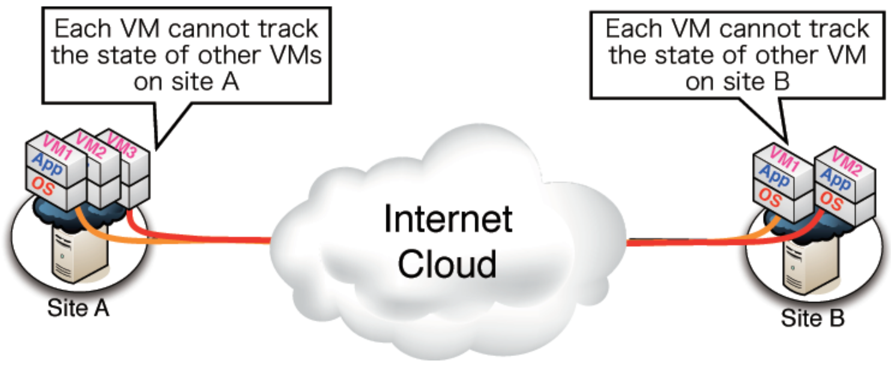

Each VM cannot track the state of other VMs on site A
Each VM cannot track the state of other VM on site B
VM1
VM2
VM3
App
OS
Internet Cloud
Site A
Site B

FIGURE 1.2: Case of connection state on virtualized environment.

We present the composition of network bandwidth. The bandwidth can be divided into following criteria. Total capacity consists of available bandwidth and utilized bandwidth. The utilized bandwidth is the rate of existing flows. The available bandwidth is the maximum rate of a new flow that will not reduce the rate of existing flows. Normally, user datagram protocol (UDP) can use the available bandwidth fully. Network throughput is the average rate of successful data over a flow. Transmission control protocol (TCP) is one of mechanisms to measure the throughput. The bandwidth is shown in Figure 1.3. Again, the goal of this research is to predict the network throughput.

Here, we introduce how to predict the throughput. At first, we simultaneously generate a small transfer using TCP as a probe transfer and a large transfer using TCP as a data transfer to measure the throughput on the Internet. Next, the measurement results are used as historical data. In other words, such prediction can be formulated as a regression problem. Finally, we apply statistics and machine learning techniques [13]

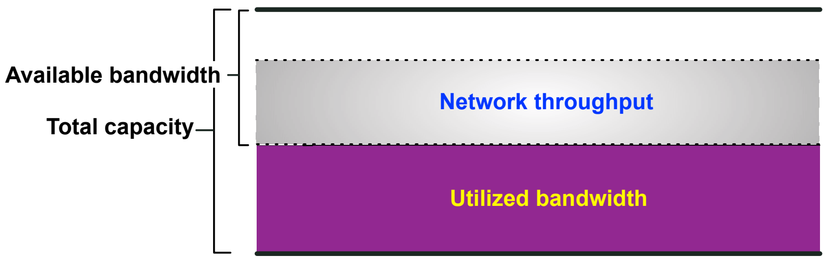


FIGURE 1.3: Composition of bandwidth on end nodes.

to the results to determine an appropriate regression curve. To summarize, we focus on the appropriate regression curve using the past measurements. The prediction problem is described in Figure 1.4.

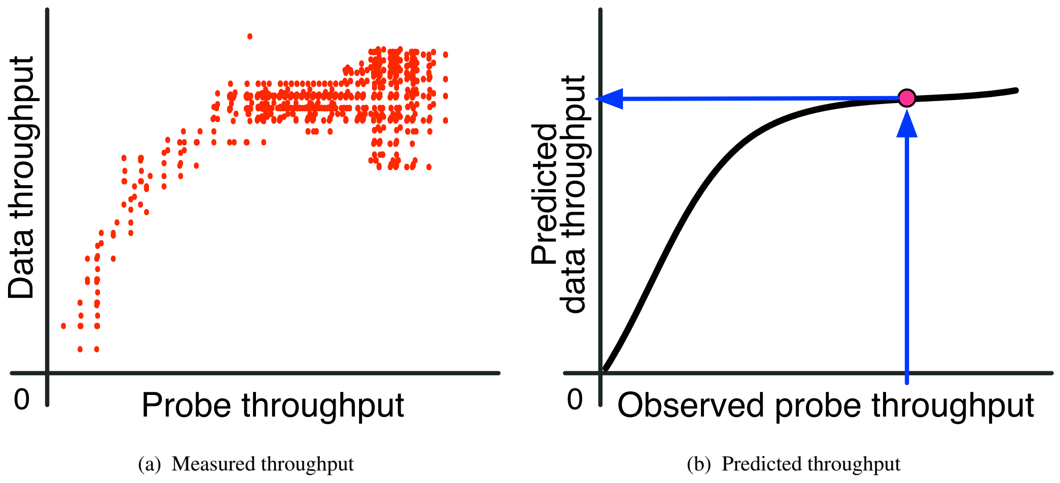


FIGURE 1.4: Problem of network throughput prediction.

There are research questions for the throughput prediction. We will attempt to answers the following research questions:

1. What traffic characteristics are observed when the virtualization technology is used on the current Internet?
2. During network measurement, how can the impact of virtualization be estimated on application layer?
3. Will there be linearity between probe and data transfers or what other characteristics are implied for the throughput prediction?
4. What is an appropriate probe size for precise prediction?

5. What algorithms are proper to determine an appropriate fitting curve on noise and non-linear characteristics?

6. How can prediction results contribute applications? Is it true that precise prediction results should always guarantee the reduction of processing time when tasks are given?

1.4 Characteristics of Thesis

In order to obtain the research goals, this thesis uses non-parametric statistics and machine learning techniques. We classify the thesis as two positions of research areas; the former is a network research area and the latter is machine learning.

1.4.1 Networking

In order to predict the network throughput using historical data, there are two approaches: passive measurement and active measurement. The passive measurement only uses existing traffic, thus nothing extra traffic is occurred into networks. NetFlow [14] is one of representative passive measurements. In contrast, the active measurement uses packets that measure networks. For example, the ping program sends an ICMP packet [15], and together with its reply measures RTT on end nodes. Our approach is based on the active measurement. We simultaneously generated the probe and data transfers to measure the network throughput.

Our approach is different from previous prediction methods [6][16][17]. They focused on simply remember of past measurements [6], linear regression [16] based on Pearson product-moment correlation coefficient, or cumulative distribution function (CDF) [17]. Meanwhile, our method uses machine learning techniques to analyze massive data and to find patterns or characteristic features that help the prediction on the data.

1.4.2 Machine Learning

At aspects of machine learning techniques, the data of network traffic has many noises and the distribution of traffic fluctuation is unclear. Thus, it would be hard to determine a model of probability distribution for the prediction. Moreover, changes in data are very large because the scale and bandwidth of networks are rapidly increased each year. Machine learning techniques are appropriate indicators to take into account such data. Although our method was not a first method to apply machine learning techniques into the throughput network, it is simpler than a previous prediction method [18].

1.5 Contributions of Thesis

In this section, a list of contributions is given briefly. Although detailed descriptions of this research will be presented through Chapters 2 and 6, major contributions are outlined below.

- We show that *oversize packet spacing*, which can be caused by CPU scheduling latency, is a major cause of throughput instability on the Internet even when no significant changes occur in the well-known network metrics.
- To estimate the major cause of throughput instability, we propose two estimation methods: the former focuses on CPU availability and the latter is based on a nature of resource states with principal component analysis (PCA).
- We select an appropriate probe size for the prediction model through statistics approach. We make clear that a previous proposed probe size did not have high predictability and it was not enough to real applications.
- A network throughput prediction method with improved accuracy is proposed. The prediction results of the proposed method are more accurate and robust than the previous method [17] for the same data sets.
- Through a meta-scheduler simulation, we clarify how high accuracy of throughput prediction should contribute to realize efficient scheduling. Next, we show that the schedulers with throughput predictors should not always guarantee the reduction of processing time on give tasks. Only a few of large prediction errors can drastically affect the overall processing time. As a result, it takes more time than the scheduler without any predictions although precise prediction results reduce the processing time.

1.6 Thesis Organization

In this thesis, we investigate Internet traffic characteristics, describe a network throughput prediction method with improved prediction results, and clarify the effect of prediction results. This section illustrates thesis organization (Figure 1.5). In Chapter 1, the nature of research problems has been established and the research issues have been introduced.

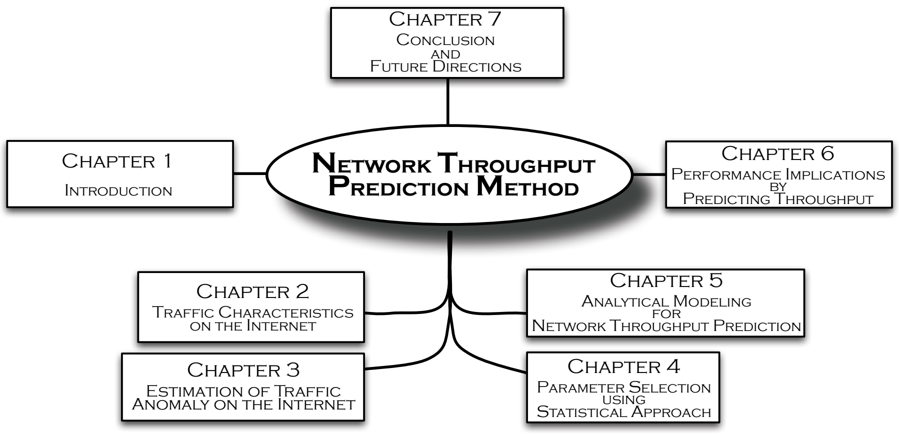

Chapter 7
Conclusion and Future Directions
Chapter 1
Introduction
Network Throughput Prediction Method
Chapter 6
Performance Implications by Predicting Throughput
Chapter 2
Traffic Characteristics on the Internet
Chapter 5
Analytical Modeling for Network Throughput Prediction
Chapter 3
Estimation of Traffic Anomaly on the Internet
Chapter 4
Parameter Selection using Statistical Approach

FIGURE 1.5: Thesis Organization.

We take an analytical approach to predict the throughput. Our approach for the prediction is presented through Chapters 2 and 5. Chapter 2 presents some studies related to network measurement with the virtualization technology to diagnose research issues of traffic characteristics. We here introduce PlanetLab [19] as a virtualized network testbed on the Internet and our throughput measurement method. We also show throughput measurement results and analysis results in packet-level to understand traffic characteristics on the Internet. In this chapter, we describe traffic characteristics [20][21][22] on the current Internet environments, clarify causes of negative impacts [20][21] of traffic by the virtualization technology, and propose a naive method [21] to get rid off the negative impacts. Moreover, the problem of naive method is clarified and an alternative estimation method [23] using principal component analysis (PCA) is described in Chapter 3. Chapter 4 is mostly concerned with the selection of prediction criteria [24] based on analysis results of Internet traffic. In this chapter, we present how to select prediction criteria through statistics approach and show throughput measurements with the criteria. Chapter 5 primarily focuses on modeling [25][26] for the throughput prediction with the analysis results in Chapter 2 and the prediction criteria in Chapter 4. Our approach is given to predict the throughput, and machine learning techniques are also introduced to determine an appropriate regression curve. In particular, prediction results are compared with a previous prediction method for the same data sets. Until this chapter, we are absorbed in our throughput prediction method, thus these chapters are essential elements to propose a precise throughput prediction method.

Chapter 6 focuses on the impact of prediction results to the grid applications. In this chapter, a simulation model of meta-scheduler using real traces of throughput is presented. We introduce some studies related to grid schedulers and services to understand research issues of grid scheduler. We present the simulation model to make clear the effect of prediction results. We show simulation results through statistics, and discuss the results. The main emphasis throughout the simulation is how significantly the high prediction accuracy contributes to the reduction of processing time.

Chapter 7 gives critical and concise summaries of the thesis, draws thesis conclusion, and describes potential future extensions.

Chapter 2

Traffic Characteristics on the Internet

2.1 Overview

In this chapter, we introduce and discuss research issues of Internet traffic characteristics when the virtualization technology is used. We first describe some studies related to network measurement on virtualized environments and a concept of packet spacing in Sections 2.2 and 2.3. Second, we present PlanetLab as a virtualized network testbed and our throughput measurement methodology in Section 2.4. In detail, we show how to select node and how to measure the throughput on the virtualized testbed. Third, throughput measurement results are described theoretically and statistically in Section 2.5. Next, we clarify a major cause of throughput instability in packet-level analysis and anomalous case is defined in Section 2.6. Moreover, a naive method for estimating the anomalous case is proposed in Section 2.7. Then, we present traffic characteristics on public cloud service in Section 2.8. Finally, we conclude with a summary of the main points in Section 2.9.

2.2 Precise Network Measurement on Virtualized Environments

In order to apply VMs to network measurement and experiments, their performance has been investigated and improved by many researchers and developers. Here, we introduce some studies related to their performance. In [28], they introduced a problem of network performance on Xen, and discussed the problem. However, they did not propose their original solution. Next, Kangarlou et. al [29] reported the problem occurred by the impact of virtualization. When many VMs shared CPU resources, CPU scheduling latency is significantly increased. It disturbed TCP transmissions to the VMs. Moreover, they proposed vSnoop, where the driver domain of a host acknowledges TCP packets on behalf of the guest VMs. Their prototype designed and implemented on Xen and they showed the improvement of throughput. The problem that addresses them is same to this research, but our approach is different from their approach. We focus on empirical-statistical criteria and methods on userland because the virtualization technology used in the provided testbeds is hardly replaceable.

In PlanetLab, a few studies [30][31] have investigated the impact of virtualization for precise network measurement and have understood Internet traffic characteristics on the virtualized testbed. Peterson et al. [30] deployed a packet forwarding overlay between Seattle and Washington, D.C on the virtualized testbed and used ping packets to compare the RTT between the Seattle and D.C. nodes for the network and overlay. The RTT of network was constant while that of the overlay varied widely. The cause of the fluctuation in RTT was CPU scheduling latency. Although CPU scheduling latency at a node will be a serious problem for network applications, no consideration has been given to the relationship between packet spacing fluctuation and scheduling latency. Spring et al. [31] showed that the load prevents accurate latency measurement and precise spacing for packet trains. They ran traceroute and tcpdump in parallel to acquire timestamps between the application and kernel levels and showed the differences between application and kernel-captured timestamps when sending probes and receiving responses. Moreover, they transmitted packet trains to determine how the CPU load impaired precisely-spaced packets. However, they showed no clear conditions for the types of load.

2.3 Packet Spacing

We here look at a concept of packet spacing and explain an ordinary value of packet spacing. We also describe the reason why methods that estimate oversize packet spacing on userland are needed.

2.3.1 Concept of Packet Spacing

A *round-trip time (RTT)* is the time required for a packet to travel from a specific source to a specific destination and back again. However, a *packet spacing* is different from the RTT. It is amount of time between the reception of a packet and the sending of the next packet. The concept of packet spacing is described in Figure 2.1. Network measurement in virtualized environment can be affected by the packet spacing in comparison with the case in non-virtualized environment. In other words, we should understand the effect of packet spacing for the precise prediction. The packet spacing is approximately 0.000010 [s] over non-virtualized environments. Although the packet spacing is an important criterion to network measurement, it is hidden to users, thus privilege or the modification of kernel is needed to measure the packet spacing.

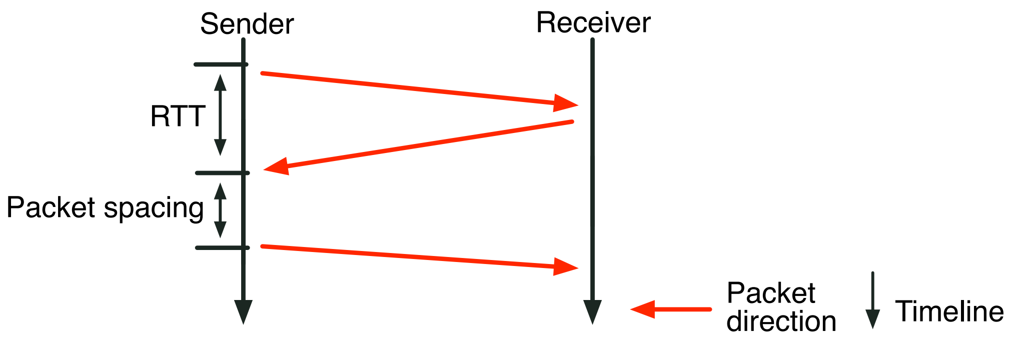


FIGURE 2.1: Concept of packet spacing.

2.3.2 Necessity for Estimating Oversize Packet Spacing on Userland

Virtualized network testbeds become popular to make network experiments with ease. In system construction on non-virtualized environment, we can re-implement lower layers, such as TCP/IP, on the system. However, the situation on the virtualized testbeds is different from the non-virtualized environment. Thus, the virtualization technology used in the provided testbeds is hardly replaceable. Moreover, the lower layers on the virtualized testbeds are normally hidden and virtualized. In other words, real resource state on a physical node is hidden and it is hard to track user behaviors on the same node over the virtualized testbed. Empirical-statistical criteria and methods that pick out oversize packet spacings for precise network experiments are required on userland.

2.4 Measurement Methodology

In this section, we first introduce PlanetLab and its characteristics as a virtualized network testbed on the Internet. Next, we present how to select node pairs before the introduction of throughput measurement method. Then, their node characteristics and mean RTT using ping are shown. Finally, we explain our throughput measurement method, and describe TCP parameters, the number of connections, and so on. It is very important to understand the measurement method because it is based on our prediction model.

2.4.1 PlanetLab as a Virtualized Network Testbed

With the progress of virtualization technology, network testbeds, e.g., PlanetLab [19][27], Emulab [32], and StarBED [33] have been widely used for network researches, distributed system researches, and network experiments. In such testbeds, resources, such as CPU, memory, and I/O interfaces, are shared and virtualized to maximize node utility for many users. PlanetLab, which is a virtualized network testbed over the Internet, has been used as a testbed for donation-based grids [34]. Moreover, it also used to investigate the validity of measurement tools [35] and prediction methods [36]. We have found that this kind of prediction needs precise measurements to obtain the learning samples. As of February 2013, it has grown to 1,100 machines spanning more than 500 sites and 40 countries. In PlanetLab, a node based on the Linux-VServer [11] shares the resources. A *sliver* represents a collection of resources assigned to a user on a physical node. Thus, the sliver is a virtualized environment on the physical node. Multiple slivers can be run simultaneously at each node. A set of these slivers participating in the same activity at different nodes is called a *slice*. Thus, PlanetLab consists of virtualized nodes on the Internet.

For a better understanding of PlanetLab, we prepare an example of PlanetLab architecture on the Internet (Figure 2.2). There are four types of slivers and five PlanetLab nodes on the Internet. The green slivers on nodes A, C, D, and E are connected on the Internet. In other words, the green slice consists of the green slivers. While users of green slice cannot access node B, they can access the nodes A, C, D, and E.

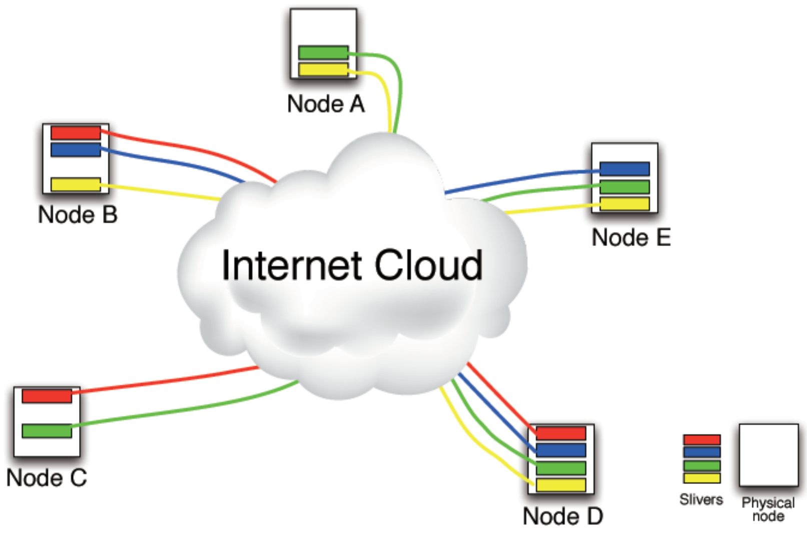


FIGURE 2.2: PlanetLab architecture.

2.4.2 Node Selection

To investigate the traffic characteristics, we empirically select four node pairs (eight nodes), which we refer to these pairs as (α, β), (γ, δ), (κ, λ), and (μ, ν). The nodes are contributed by universities or research institutions across North America and Europe participating to PlanetLab, and composed of two or four independent CPU cores and physical memory of approximately 3 GB (gigabytes). The virtualized network testbed does not support network virtualization [37][38] for virtualized networks, thus the resource is shared among slices. It would be possible to use it at each site, but we did not consider it in the experiments. We measure the throughput using Iperf and RTT using ping; their basic characteristics are given in Table 2.1. We observed throughput instability and abrupt RTT fluctuation in node pairs for two pairs (κ, λ) and (μ, ν). To show the abrupt RTT fluctuation, we introduce RTT using ping at pair (κ, λ) in Figure 2.3. We find an abrupt value of RTT (0.3990 [s]), and it is larger than the others.

TABLE 2.1: Node characteristics on virtualized network testbed.

Node name	**CPU cores**	**Memory [GB]**	**Node pair** (arrow denotes transfer direction)	**Mean RTT [s]**
α	2	2.96	$\alpha \leftarrow \beta$	0.0428
β	4	3.46		
γ	4	3.42	$\gamma \leftarrow \delta$	0.0830
δ	4	3.21		
κ	4	3.42	$\kappa \leftarrow \lambda$	0.0200
λ	2	3.47		
μ	2	3.45	$\mu \leftarrow \nu$	0.0900
ν	2	2.97		

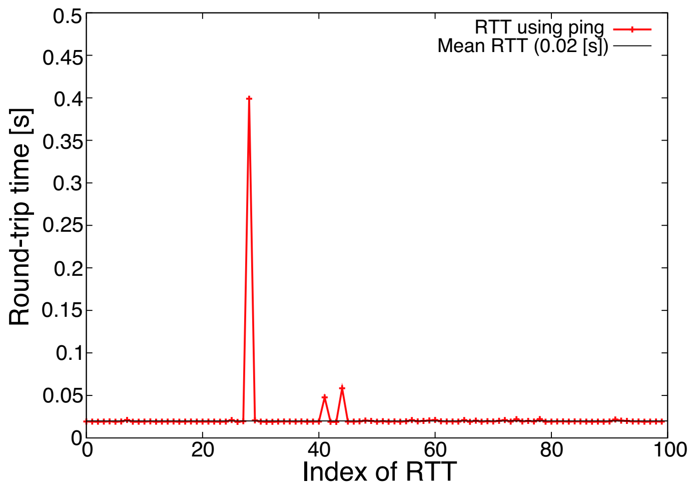


FIGURE 2.3: RTT using ping at pair (κ, λ).

2.4.3 Connection pair

A *connection pair* is a pair of probe and data transfers. The probe transfer is used for the throughput prediction and the data transfer is used for actual throughput. In other words, the connection pair consists of two simultaneous TCP connections in different sizes. The probe size is relatively smaller than the data size. The measurement results of connection pair are used for our prediction method.

We briefly introduce the previous work [16] related a prediction method using the probe transfer. Wolski et al. [16] used different-sized pairs of connections and empirically established the basic probe size as 64 KB (kilobytes) for Network Weather Service (NWS). However, they did not consider an appropriate size for probe transfer on the current Internet. We will discuss an appropriate probe size for the connection pair in Chapter 4.

2.4.4 Throughput Measurement using Connection Pair

Our objective in measuring the throughput using the connection pair is to identify Internet traffic characteristics for network throughput prediction on network virtualization areas [39][40]. We used multiple TCP connections to press router's queue at end-to-end path and to encounter congested network state. In [41], when the number of connections was over 8 on non-virtualized testbed, the throughput was saturated. Moreover, Altman et al. [42] found that aggregate throughput is not saturate on the virtualized testbed when the number of connections is below 5. In the virtualized testbed, there are various kinds of network traffic of the other slices on the node. We considered the above situation, and used 6 connections (3 connection pairs) simultaneously. We generated the various sizes for the connection pair every 5 minutes at all the pairs, and monitored the resource state per slice on the node with slicestat [43] every 1 minute during throughput measurement. If the measured size is smaller than expected or if the transfer time is more than 5 minutes, we judge that the experiment has failed. Advertised window sizes for probe and data transfers are respectively reduced to 16 and 64 KB. These window sizes for the connection pair have been based on NWS. The sizes for the connection pair are based on the original sizes and shown in Table 2.2. The measurement methodology is described in Figure 2.4.

TABLE 2.2: Probe and data sizes for connection pair.

Index	**Probe size [KB]**	**Data size [MB]**	**Number of connection pairs**
1	16	8	3
2	16	16	3
3	32	16	3
4	64	16	3
5	64	32	3

Although CPU scheduling latency [30] and heavy loads [31] are likely to affect the network measurement, the impact of virtualization over many sites was not investigated and no clear conditions or criteria had been established to estimate unstable conditions. Moreover, it is hard to directly observe CPU scheduling latency on the virtualized network testbed. Our approach is to implement a simple CPU monitoring program that gets the current time every second to investigate CPU availability. It consists of a loop for timestamp acquisition. In each iteration, the monitoring program calls get-timeofday() and the timestamp is saved in a pre-allocated memory. If the resource state is stable, the constant spaced timestamps are stored and one CPU core will be allocated to the monitoring program fully. It is run during throughput measurement.

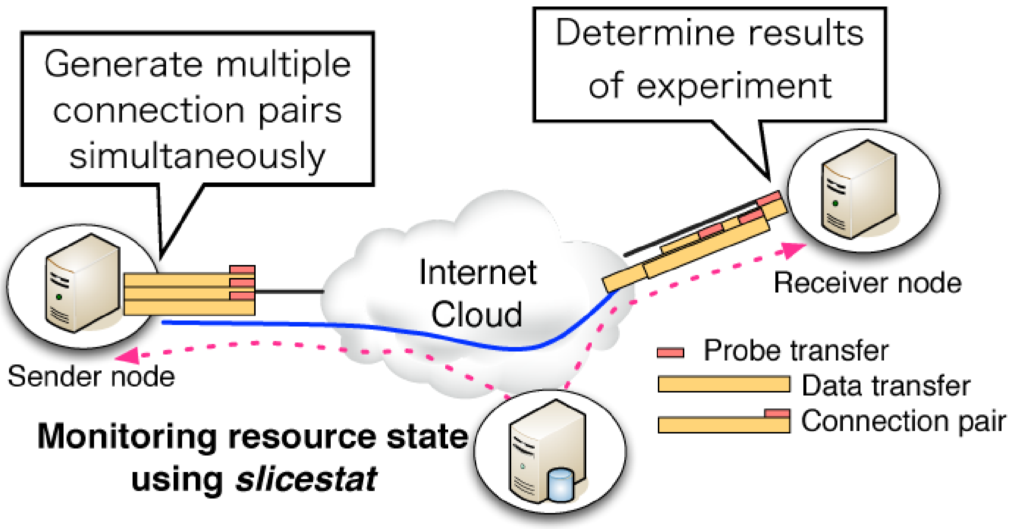

Generate multiple connection pairs simultaneously
Determine results of experiment
Internet Cloud
Receiver node
Sender node
Probe transfer
Data transfer
Connection pair
Monitoring resource state using *slicestat*

FIGURE 2.4: Connection pair measurement method with resource monitoring.

2.5 Throughput Measurement Results

In this section, we first present a formulation that calculates theoretical throughput. It is based on parameters in TCP headers. Next, we show actual throughput for the connection pair. Finally, we summarize the measurement results with the statistics.

2.5.1 Theoretical Achievable Throughput

Theoretical achievable throughput is defined as

$$Throughput = \frac{W \cdot p}{P \cdot R} \tag{2.1}$$

where W is advertised window size, p is packet size without TCP/IP headers, P is packet size with headers, and R is RTT. In the theoretical achievable throughput, we assume that there is no packet loss and advertised window is saturated to the maximum size. For example, when the advertised window is 64 KB, RTT is 0.02 [s], packet size without headers is 1368 Bytes, and packet size with headers is 1420 Bytes, the ideal case using the theory is 3156.7 KBps. We can compare differences in the ideal and the measurement results.

2.5.2 The Statistics of Measurement Results

We gathered a data set of approximately 2,000 connection pairs from all the node pairs everyday. Table 2.3 presents the ideal case using the theoretical achievable throughput and the statistics when the probe size was 64 KB and the data size was 32 MB. The parameters, such as packet size, header size, and so on, of the ideal are equal to the measurement results. The mean throughput at pairs (α, β), (κ, λ), and (μ, ν) is different from the ideal case. Moreover, the maximum throughput at pairs (κ, λ) and (μ, ν) is not close to the ideal case. It is very different in comparison with the maximum value at pairs (α, β) and (γ, δ). Moreover, we find a significant difference in the statistics at pairs (κ, λ) and (μ, ν). Figure 2.5 shows throughput measurement results of the above size. Also, the decreases in the throughput are observed on two pairs (κ, λ) and (μ, ν).

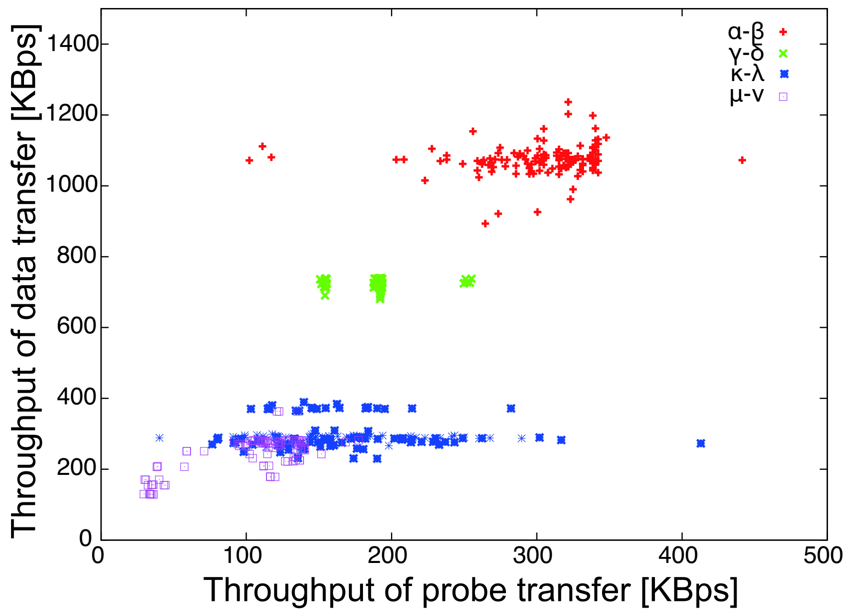


FIGURE 2.5: Throughput measurement results (probe: 64 KB, data : 32 MB).

TABLE 2.3: Ideal, minimum, mean, and maximum throughput.

Node pair	**Ideal [KBps]**	**Min [KBps]**	**Mean [KBps]**	**Max [KBps]**
α - β	1475.1	893.4	1073.7	1236.5
γ - δ	760.6	680.4	730.6	739.2
κ - λ	3156.7	251.9	288.6	413.1
μ - ν	701.5	128.7	241.9	363.3

2.6 Packet-level Analysis

In this section, we describe packet-level analysis. Our objective in packet-level analysis is to find the cause of throughput instability and to clarify the Internet traffic characteristics. We analyze well-known network metrics, such as RTT and packet loss rate, for the connection pair where the probe size was 64 KB and the data size was 32 MB in all the node pairs. Next, we investigate packet spacing and advertised window at the sender. Finally, we introduce anomalous cases to understand traffic characteristics on the virtualized testbed.

2.6.1 Round-trip Time (RTT)

The RTT normally fluctuates and the throughput decreases as a result of network congestion on the end-to-end path. Table 2.4 shows the mean RTT for all the node pairs. We find increases in the mean RTT at pair (α, β) in comparison with the mean RTT using ping. However, the range of fluctuation of the mean RTT at the others is smaller than the case at pair (α, β). To investigate the fluctuations in RTT, we present the cumulative distribution function (CDF) of RTT in Figure 2.6. The values of throughput at all the pairs (α, β), (γ, δ), (κ, λ), and (μ, ν) are 1085.8 KBps, 738.3 KBps, 399.0 KBps, and 363.3 KBps respectively. In the CDF at pair (α, β), RTT is gradually increased, and it is a cause of fluctuation in throughput. Although the approximate 90% of RTT is close to the mean RTT using ping at two pairs (κ, λ) and (μ, ν), the decreases in the network throughput are occurred. Thus, the fluctuations in RTT are not enough to prove the major cause of throughput instability at two pairs (κ, λ) and (μ, ν).

TABLE 2.4: Mean RTT of connection pair.

Node pair	**Mean RTT Probe [s]**	**Mean RTT Data [s]**	**Mean RTT Ping [s]**
α - β	0.0520	0.0556	0.0428
γ - δ	0.0854	0.0829	0.0830
κ - λ	0.0224	0.0225	0.0200
μ - ν	0.0974	0.0974	0.0900

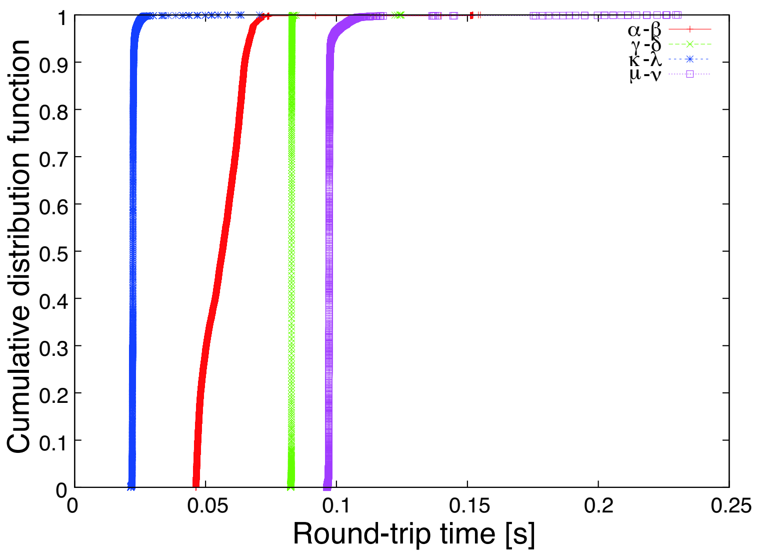


FIGURE 2.6: CDF of RTT of data transfer at all node pairs (32MB).

2.6.2 Packet Loss Rate

The packet loss rate is normally an important metric for the throughput. If it is high, the network throughput is decreased. The mean loss rates at all the pairs are shown in Table 2.5; they are smaller than 1%. To summarize, both packet loss and the fluctuations in RTT were the major cause of throughput fluctuation at pair (α, β). Conversely, these metrics could not clarify the cause of throughput instability at two pairs (κ, λ) and (μ, ν). Thus, the cause of throughput instability at two pairs (κ, λ) and (μ, ν) was not any network effect, such as network congestion, on the virtualized testbed. Moreover, we find that there is no packet loss though the values of RTT are increased. In it, the mean RTT is 0.0231 [s], the mean packet spacing is 0.013366 [s], and the throughput is 283.5 KBps. To identify the validity of the above case, we investigate TCP's timeout interval [44] of the case. We present RTT, estimated RTT, and the timeout interval of the above case in Figure 2.7. Although the values of RTT are increased, the timeout intervals are larger than the values of RTT, and there is no packet loss.

TABLE 2.5: Mean packet loss rate of connection pair.

Node pair	**Mean loss rate [%] (Probe)**	**Mean loss rate [%] (Data)**
α - β	0.032	0.049
γ - δ	0.216	0.015
κ - λ	0	0.001
μ - ν	0	0.001

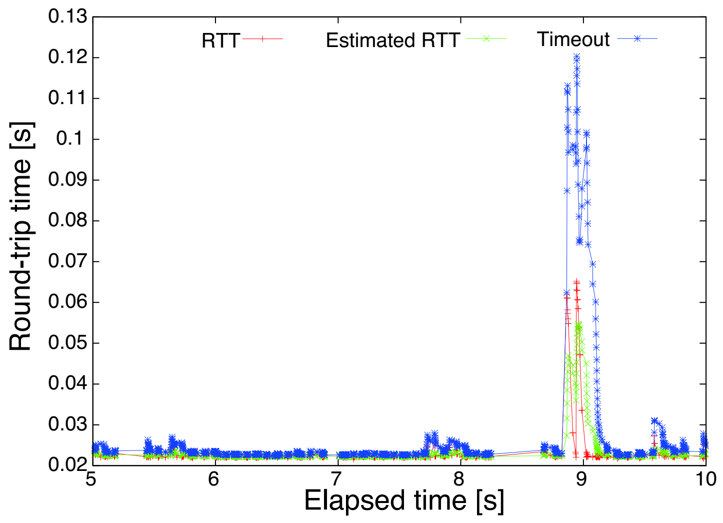
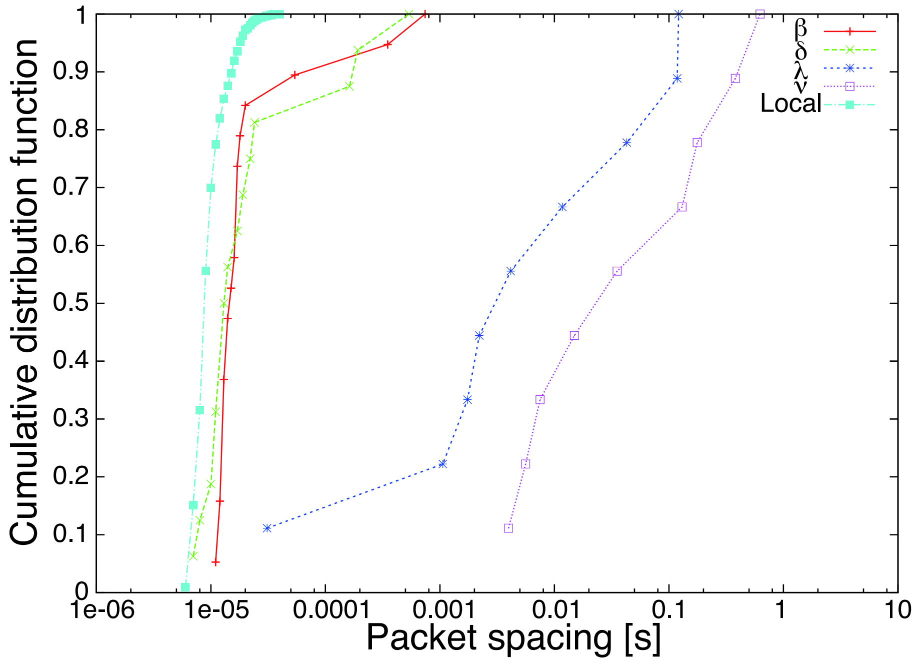


FIGURE 2.7: RTT, Estimated RTT, and Timeout of data transfer at pair (κ, λ).

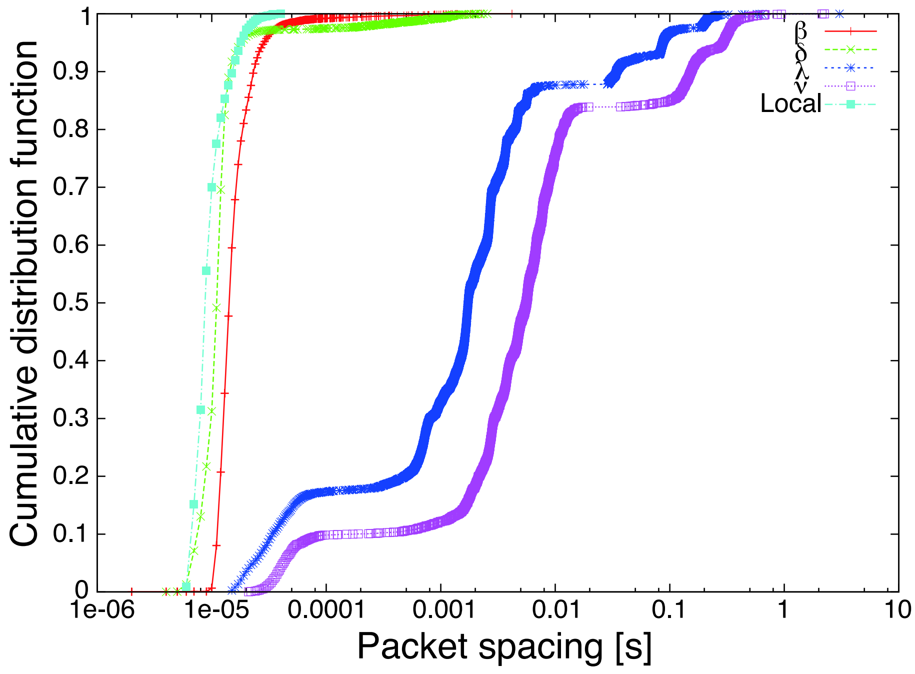
2.6.3 Cause of Network Throughput Drop

We observed decreases in the network throughput for particular node pairs. When many connections exceed the acceptable number of connections concurrently concentrated on the receiver node, the network throughput decreases. Rak et al. [45] observed the number of pending requests at a server by increasing the number of requests to evaluate Web Service resilience. However, our case differs from the above case because resources, such as memory and network bandwidth, at each node are limited by a watchdog daemon. This daemon forcibly reduces a user's memory rate when the memory rate suddenly exceeds the threshold amount.

Again, the packet spacing is a very short period over non-virtualized environments. To investigate packet spacing over a non-virtualized environment, we considered that a local environment consists of a router and two native nodes. Each native node has an Intel Pentium 4 processor with a 1 GB RAM and a 100 Mbps network card and runs on Ubuntu 9.04. The router was connected between these nodes via a 100 Mbps link. There are no heavy loads at each node and no network traffic between the nodes. We generated the sizes for connection pair in Table 2.2. When data size was 32 MB, the minimum, mean, maximum values of packet spacing are 0.000006 [s], 0.000010 [s], and 0.000040 [s] respectively. In the local, there is no abrupt fluctuation in the packet spacing and these values do not affect the throughput measurement. Next, we show the packet spacing on the virtualized testbed. Figure 2.8 shows the CDF of packet spacing at the local and that for connection pair where the probe was 64 KB and the data was 32 MB at all the pairs. The approximate 80% of packet spacings for the probe transfer and the approximate 90% for the data transfer at nodes β and δ are similar to the packet spacing at the local. In the case at node β, the throughput of the data transfer is 1093.9 KBps, the mean RTT is 0.0559 [s], and the mean packet spacing is 0.000021 [s]. The CDF for connection pair at node β is similar to the CDF of the local. Although a portion of packet spacings at node β is increased, these cannot be the major cause of fluctuations in the throughput. The case at node δ is similar to the above case. However, the packet spacings at nodes λ and ν are larger than the other nodes. We introduce the CDF of packet spacing at node λ. In this case, the network throughput is 288.1 KBps, mean RTT is 0.0223 [s], and mean packet spacing is 0.014064 [s]. Although the mean RTT is similar to that using ping, a portion of the packet spacings is larger than the packet transmission period, and the CDF of packet spacing is very different from the case at the local and nodes β and δ. Moreover, we show the mean packet spacing for connection pair where the probe was 64 KB and the data was 32 MB in Figure 2.9. The mean packet spacing at nodes λ and ν is larger than the case at nodes β and δ, and these are very different from the case at the local.



(a) Probe transfer (64 KB)



(b) Data transfer (32 MB)

FIGURE 2.8: CDF of packet spacing for connection pair at all pairs.

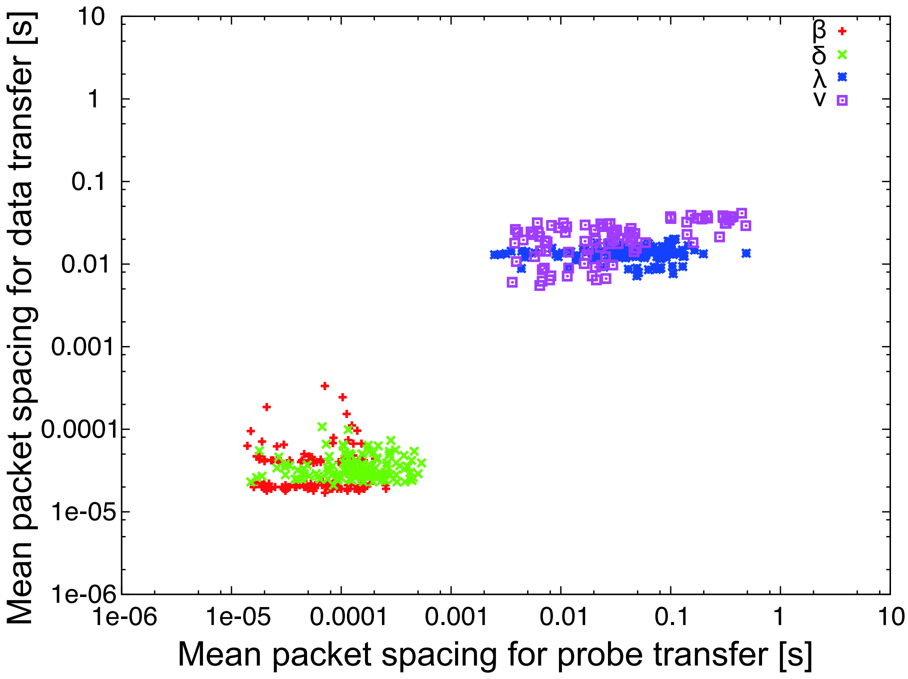
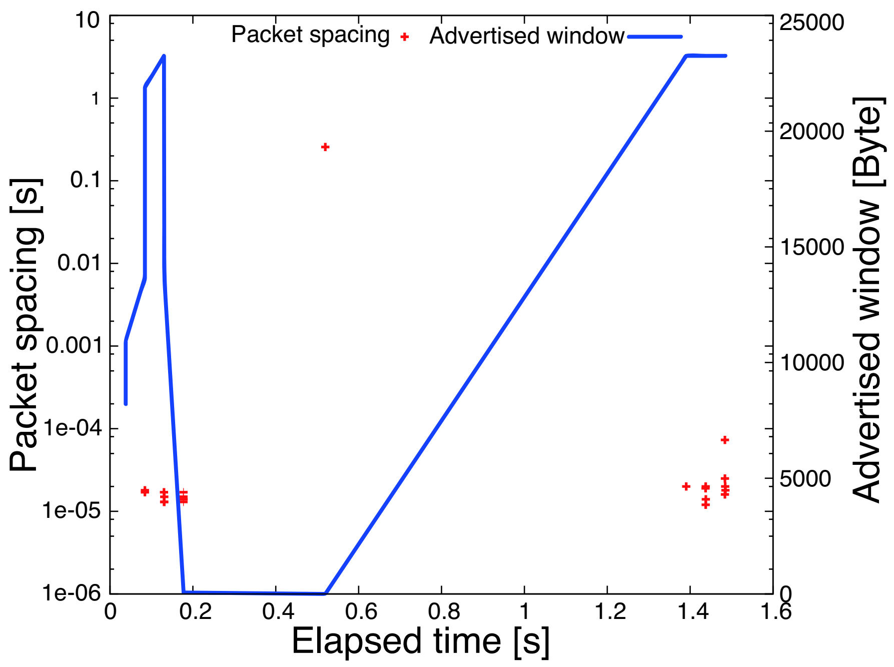


FIGURE 2.9: Mean packet spacing for connection pair at all senders (probe: 64 KB, data : 32 MB).

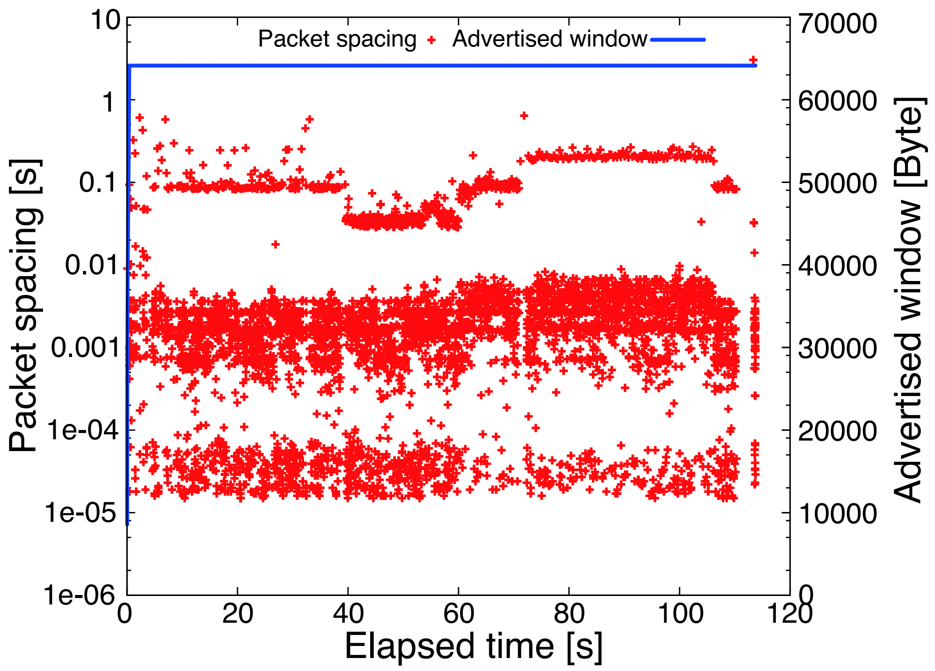
The oversize packet spacing as compared to that over non-virtualized environment was a major cause of throughput instability even when there were no abrupt changes in the well-known network metrics at pairs (κ, λ) and (μ, ν). It accords with results at the previous work [30]. These oversize packet spacings are unusual anomalies. When the packet spacing was larger than RTT, packets at the sender node were not sent consecutively. These rarely happen in non-virtualized network environment while it is easy for anomalies to be occurred in virtualized network environment. The problem described above is more severe on the virtualized environments than the non-virtualized environments because resources are virtualized and shared among virtual machines, and it would be hard to monitor the other virtual machines on the node.

2.6.4 Advertised Window

TCP provides flow control [46] for the receiver to control transmission speed, so that the receiver is not overwhelmed with data from the sender. The advertised window is used to give the sender an idea of how much free buffer space is available at the receiver. If the advertised window at the receiver is zero, the sender does not send data to the receiver after ACK packet; thus, the packet spacing is increased at the sender. We show two cases in Figure 2.10; one case is that the packet spacings are increased and the advertised window is decreased due to flow control while the other case is that the packet spacings are increased and the advertised window is saturated to up maximum size. In it, packet spacings do not relate to flow control.



(a) Packet spacing and advertised window of probe transfer at β (64KB)



(b) Packet spacing and advertised window of data transfer at λ (32MB)

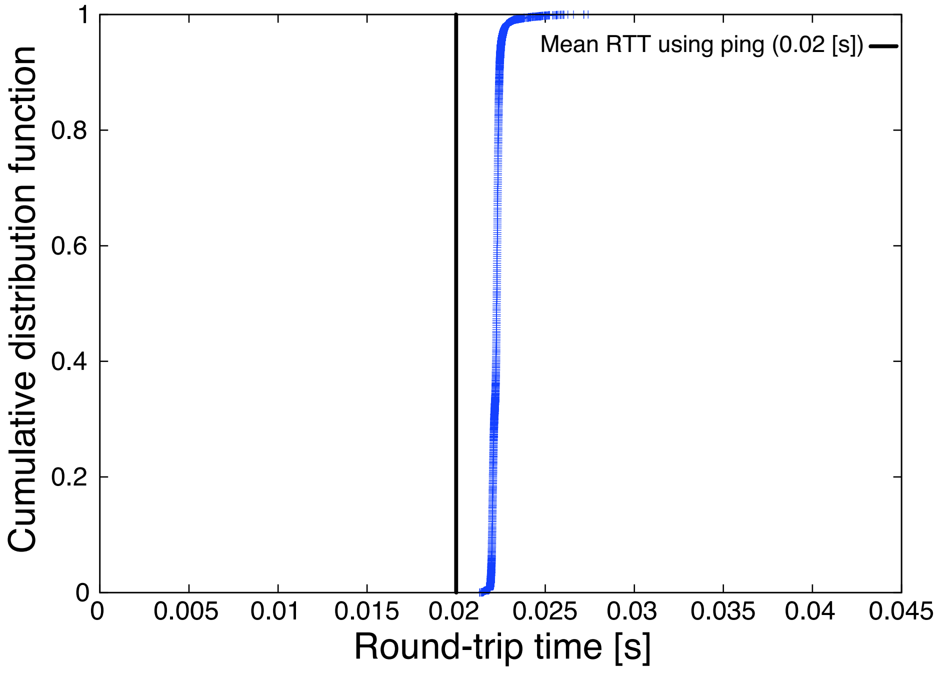
FIGURE 2.10: Packet spacing and advertised window.

2.6.5 Anomalous Case

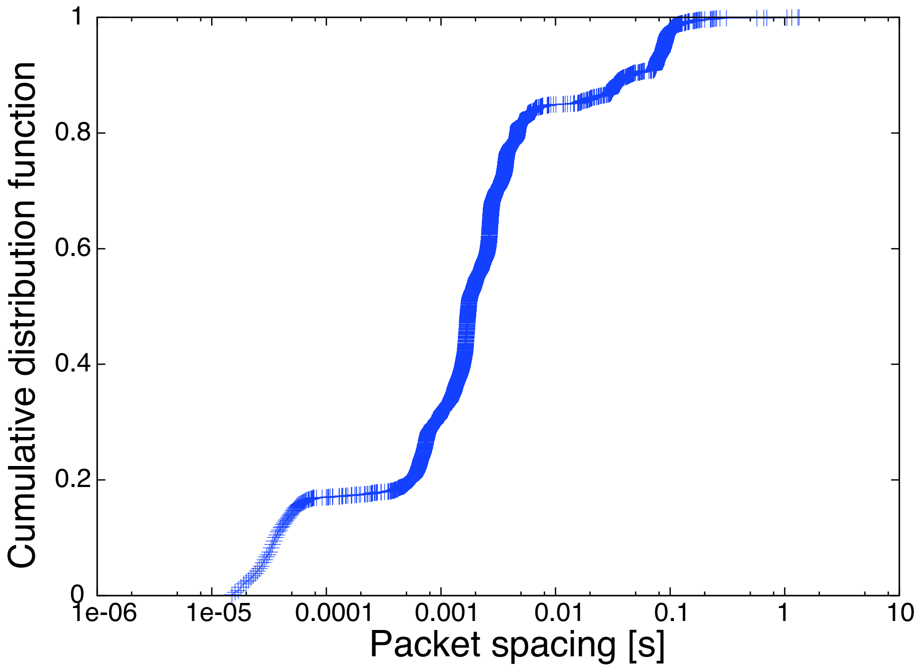
To summarize, the result at pair (α, β) experienced the fluctuations in network throughput caused by the packet loss and the fluctuations in RTT. This case is normally predictable. But, the case at two pairs (κ, λ) and (μ, ν) differed from the above case. In this case, the network throughput drop was occurred by the anomalies with stable network state.

Major distinction of anomalous cases is the throughput instability despite of stable network state, which can be observed through RTT, packet loss rate, and advertised window. The condition for the judgment of the anomalous case is the impact of packet spacing. We introduce the case at node λ as an example of the anomalous case. We show the CDF of RTT and packet spacing in Figure 2.11. In this case, the throughput is 288.6 KBps, the loss rate is zero, the advertised window is saturated to the maximum size (64 KB), and minimum, mean, and maximum RTT are 0.0213 [s], 0.0222 [s], and 0.0274 [s]. The statistics of RTT are close to the mean RTT (0.02 [s]) using ping. However, minimum, mean, and maximum values of packet spacing are 0.000015 [s], 0.013618 [s], and 1.332624 [s] respectively. The CDF of packet spacing at node λ is different from the CDF at nodes β and δ. To summarize, the most of values of RTT are stable, and the throughput instability are occurred by the anomalies. This case is the anomalous case. If the network throughput is decreased by the anomalies, we should carefully review measurement results.

Finally, we investigate the impact of the packet spacings to the throughput instability at pairs (κ, λ) and (μ, ν). To prove stable network state, we show the CDF of RTT at pairs (κ, λ) and (μ, ν) in Figure 2.12. Most of values of RTT are close to the mean RTT using ping. It is sufficient to show the stable network. Despite the stable network state, the statistics of network throughput are different from the ideal case in Table 2.3. We show the CDF of the packet spacing for the judgment in Figure 2.13. The distribution of the CDF at nodes λ and ν differs from that at nodes β and δ.

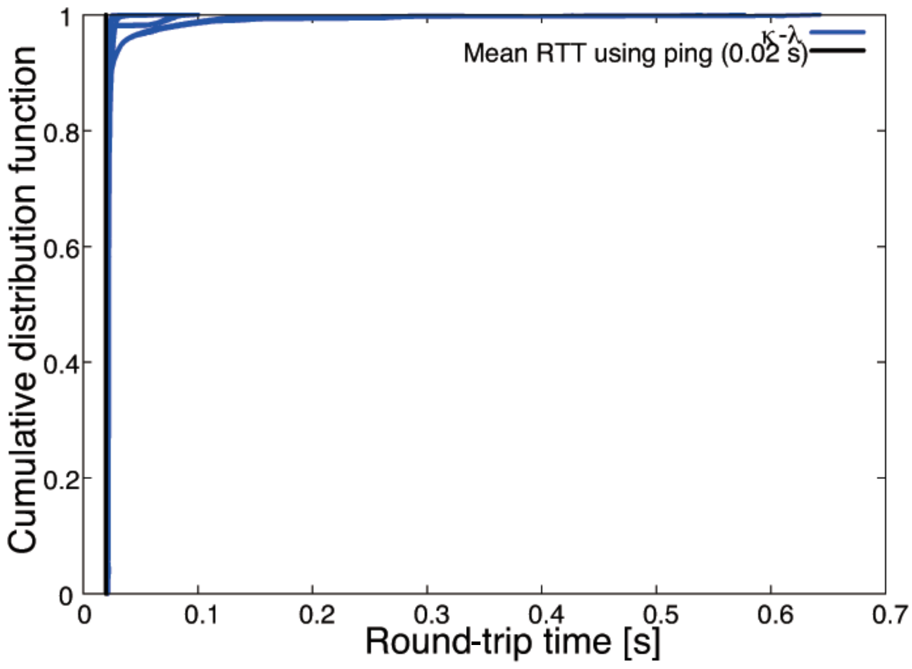


(a) CDF of RTT

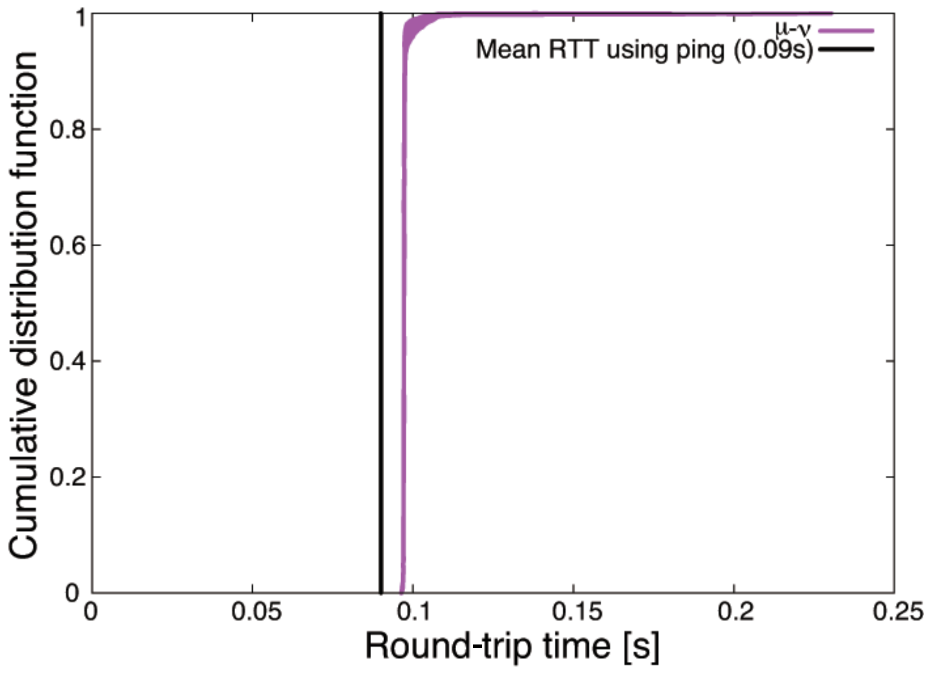


(b) CDF of packet spacing

FIGURE 2.11: CDF of RTT and packet spacing at λ (32 MB).

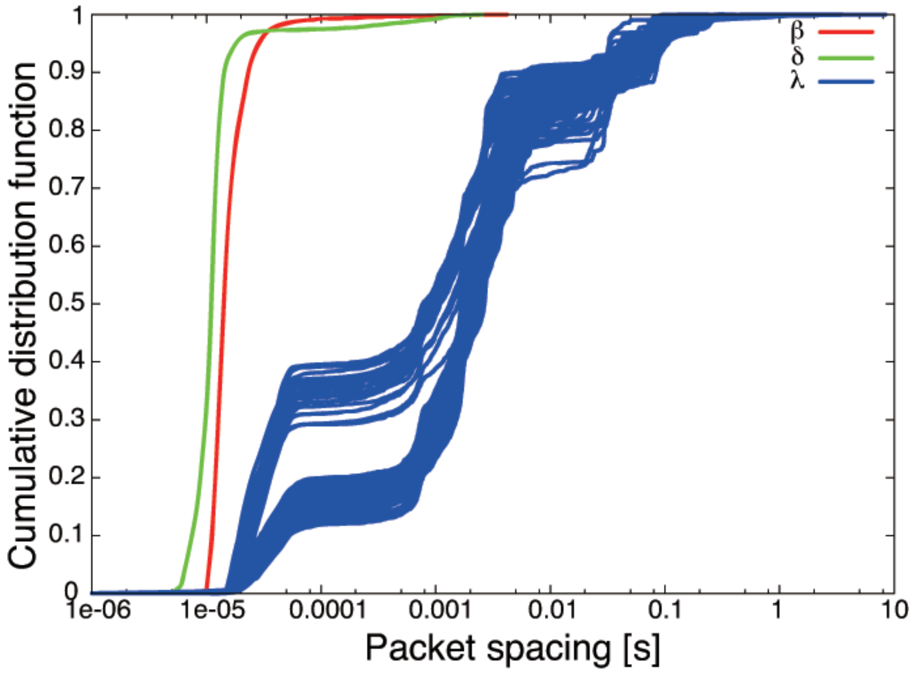


(a) CDF of RTT at κ and λ

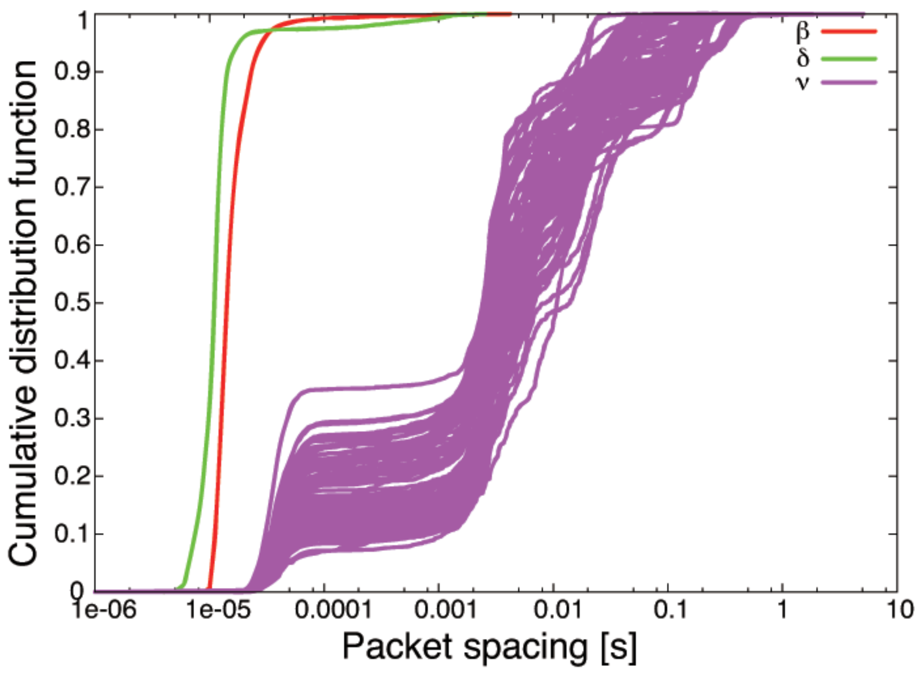


(b) CDF of RTT at μ and ν

FIGURE 2.12: CDF of RTT at anomalous case (32 MB).



(a) CDF of packet spacing at β, δ, and λ



(b) CDF of packet spacing at β, δ, and ν

FIGURE 2.13: CDF of packet spacing at anomalous case (32 MB).

2.7 Naive Method for Estimating Anomalous Case

In the previous section, we observed decreases in the throughput at the particular pairs. The major causes of throughput decrease were anomalies. Previous works [30][31] did not investigate the anomalies at many sites. Moreover, no clear conditions or criteria had been established to estimate the effect of the anomalies. During the generation of the connection pair, we analyzed results of resource monitoring to investigate the anomaly condition and to establish efficient criteria to estimate the anomalous case.

2.7.1 CPU Utilization

In a multi-core processor, if there are two or four CPU cores on the processor, maximum CPU utilization can be showed to 200% or 400%. However, these can give us confusion. We divided these values into the number of cores, and normalized these values as 100%. In order to investigate the condition of anomalous case, we observed CPU utilization at the sender node using slicestat during the connection pair generation. The CPU utilization at all the sender nodes is shown in Figure 2.14. The CPU utilization at node δ was lower than the other senders. In this case, there were no anomalies because of very low CPU utilization.

Although there was a difference in the mean packet spacings, the CPU utilization did not reflect the mean packet spacings without node δ. In this case, it will be inappropriate for estimating the anomalous case on the virtualized testbed. We show the statistics of CPU utilization at all the senders in Figure 2.15. In the statistics, the case at node β does not differ from the case at nodes λ and ν. The statistics are not enough to estimate the anomalous cases.

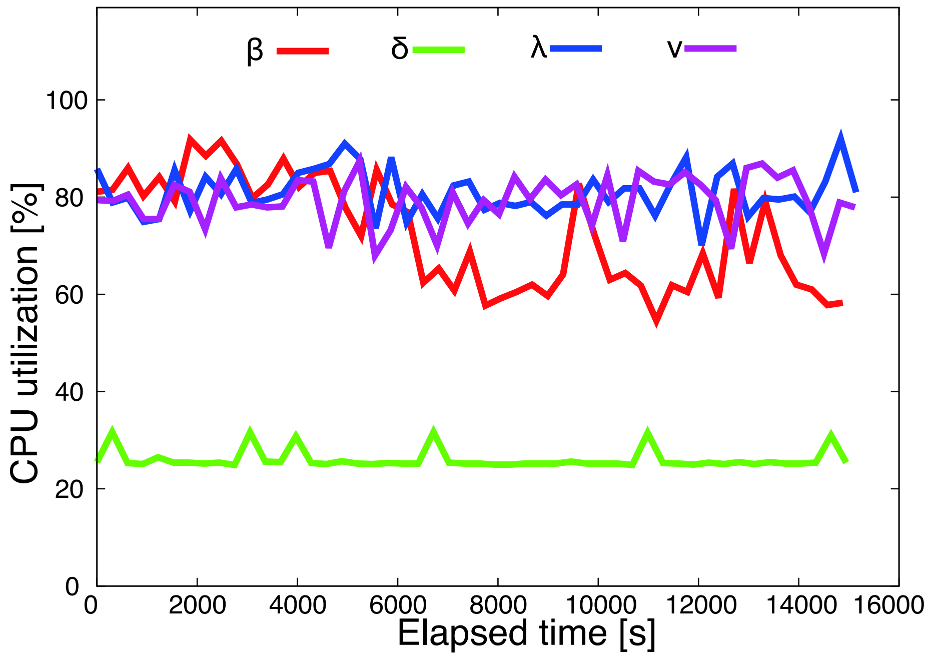


FIGURE 2.14: CPU utilization at all sender nodes.

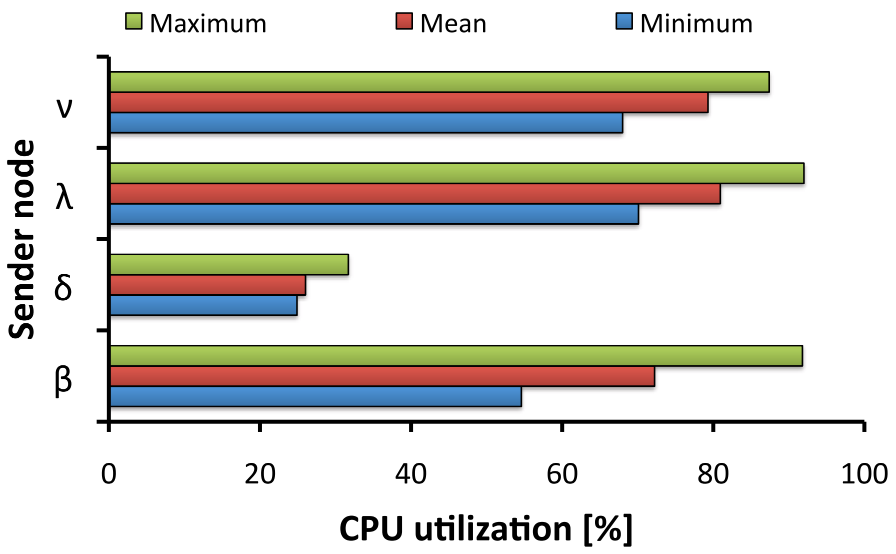


FIGURE 2.15: Minimum, mean, and maximum CPU utilization at all sender nodes.

2.7.2 CPU Availability

While we ran the monitoring program that consumes CPU cycles at the node, we observed CPU availability at the node using the monitoring program. The CPU availability is different from the CPU utilization. It is available CPUs to be allocated to users while the CPU utilization is consumed to users. If there are two CPU cores at the node and the CPUs are idle, one will be fully allocated to the monitoring program. Thus, it uses one CPU core and the CPU utilization can be increased to 50%. Conversely, it will be hard to allocate the CPU to the monitoring program if the CPUs are busy. Figure 2.16 shows the CPU availability at all the sender nodes. There is a significant change in the CPU availability. Although there were two CPU cores at nodes λ and ν, it was not increased to approximately 50%. These nodes had the anomalies. Conversely, there are four CPU cores at node δ. It is keeping up the approximately 25% and the small packet spacing. The CPU availability is an important criterion for estimating the anomalous cases on the virtualized network testbed. When the CPU utilization is high and the CPU availability is low, the anomalies are occurred.

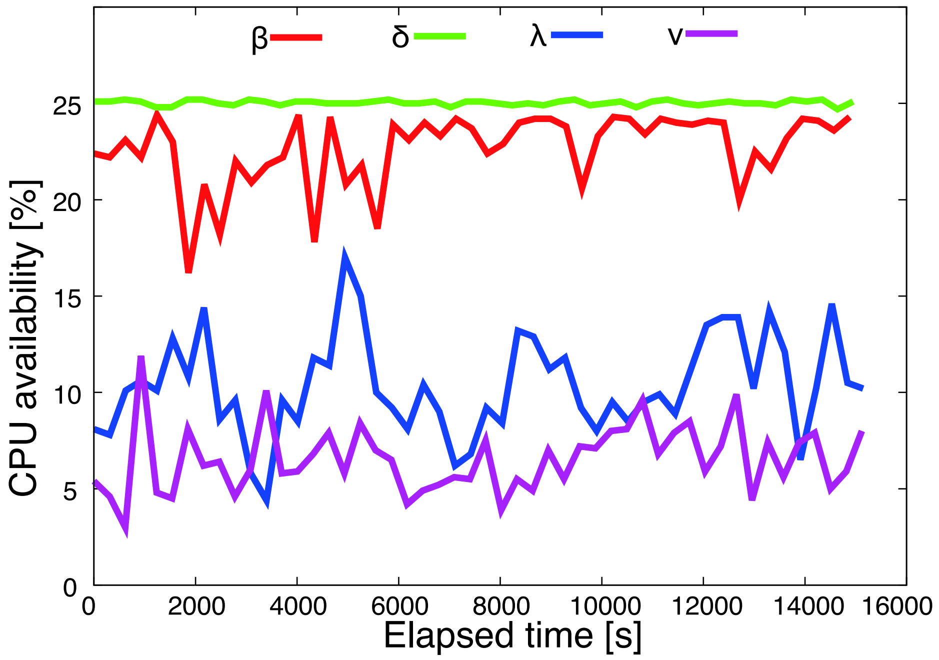


FIGURE 2.16: CPU availability at all sender nodes.

2.8 Traffic Characteristics on Cloud Service

In this section, we show the traffic characteristics on different virtualized environment. We first introduce Amazon EC2 [4] and its performance analysis results discussed in previous works. Next, our experimental environment is described for throughput measurements. Then, throughput measurement results are shown through statistic analysis. Finally, CPU resources are presented to indentify the reproduction of the anomalous cases.

2.8.1 Amazon EC2

Cloud services emerge as a new paradigm recently, and provide users to acquire computing resources from large scale data centers of service providers, such as Amazon, Google, and Microsoft. Amazon EC2 [4] is one of popular cloud services. It is a component of Amazon web services (AWS) [47], and its web service interface allows users to obtain *Amazon Machine Instance (AMI)* and to run applications on the AMIs. The AMI based Xen is provided to user as the platform. The instances e.g., micro, small, large, and so on, have different resources, such as CPU cores, memory, and storage. Amazon EC2 provides the ability to place the instances in multiple locations. The locations are composed of regions and availability zones. The zones are distinct locations that are engineered to be insulated from failures in other zones and provide inexpensive and low latency network connectivity to other zones in the same region. It is currently available in seven regions: US East (Northern Virginia), US West (Northern California, Oregon), South America (Sao Paulo), EU (Ireland), and Asia Pacific (Singapore, Tokyo).

2.8.2 Performance Analysis of Amazon EC2

We here introduce some studies related to performance analysis of Amazon EC2.

Iosup et al.[48] investigated EC2 performance using micro-benchmarks, kernels, and eScience workloads. They evaluated the instances as a scientific computing platform only. Dejun et al.[49] analyzed EC2 performance for service-oriented applications. They developed a CPU-intensive web application and database read and write-intensive applications to simulate different types of workload pattern. Despite the performance of the instances to be stable, the multiple instances of the same type showed heterogeneous performances. The performance analysis was focused, but they did not discuss traffic characteristics of instances.

To our best knowledge, Wang et al.[50] focused on networking performance among the instances, and demonstrated very different characteristics, such as abnormal large delay variations and unstable TCP/UDP throughput, caused by the impact of virtualization on node. However, they focused on the regions on Amazon EC2 data center only, and did not investigate networking performance between the region and different types of network. In these studies, they did not use monitoring programs offered by AWS. Because the lower layers on the virtual machine are block-box typically, their approach would be inappropriate for the performance analysis.

2.8.3 Throughput Measurement

We choose the micro instance as a sender node to observe the anomalies and to identify the impact of virtualization. The five micro instances are used for the experiments. We also choose a native node on our university as a receiver node to eliminate the impact of virtualization. The native node has an Intel Xeon CPU (4 CPU cores) with a 4 GB RAM and an 1 Gbps network card and runs on Ubuntu 10.04. Meanwhile, the micro instance is with 613 MB memory, 1 EC2 compute unit (Intel Xeon), and 10 GB on EBS storage. However, we cannot have information about the physical node. Then, the virtualization level is also different from PlanetLab. Amazon EC2 uses Xen, while PlanetLab uses V-Server. It is with operating system-level virtualization while the micro instance is with para-virtualization. The instances are in US East - N. Virginia (us-east-1a). Min, max, and mean RTT between the region on Amazon EC2 and our node are 0.209 [s], 0.369 [s], and 0.213 [s]. Our selection of instance is appropriate because the experiments use mainly network resources. In order to measure the actual throughput, we use the connection pair. Moreover, we prepare both small and large-sized probe for the connection pair to identify the validity of the large-sized probe on Amazon EC2. The sizes for the connection pairs are shown in Table 2.6.

TABLE 2.6: Probe and data size combinations for connection pairs.

Index	**Probe size [KB]**	**Data size [MB]**	**Number of connection pairs**
i	16	8	3
ii	16	16	3
iii	32	16	3
iv	64	16	3
v	64	32	3
vi	512	8	3
vii	512	16	3
viii	1024	16	3
ix	1024	16	3
x	2048	32	3

2.8.4 Resource Monitoring Method

If we observe resource state using monitoring tool, such as sysstat [51], the information of resource metrics will be incorrect because the lower layers are normally black-box. We used Amazon CloudWatch [52] offered by AWS to observe the resource state during the throughput measurement. It correctly provides resource metrics such as CPU utilization, disk I/O, and network traffic per the instance. However, it does not provide memory utilization, the number of the instances on physical node, network traffic on the node, and the other instance state on the node. Moreover, the minimum granularity of the metrics is limited as 1 minute. The measurement methodology is described in Figure 2.17.

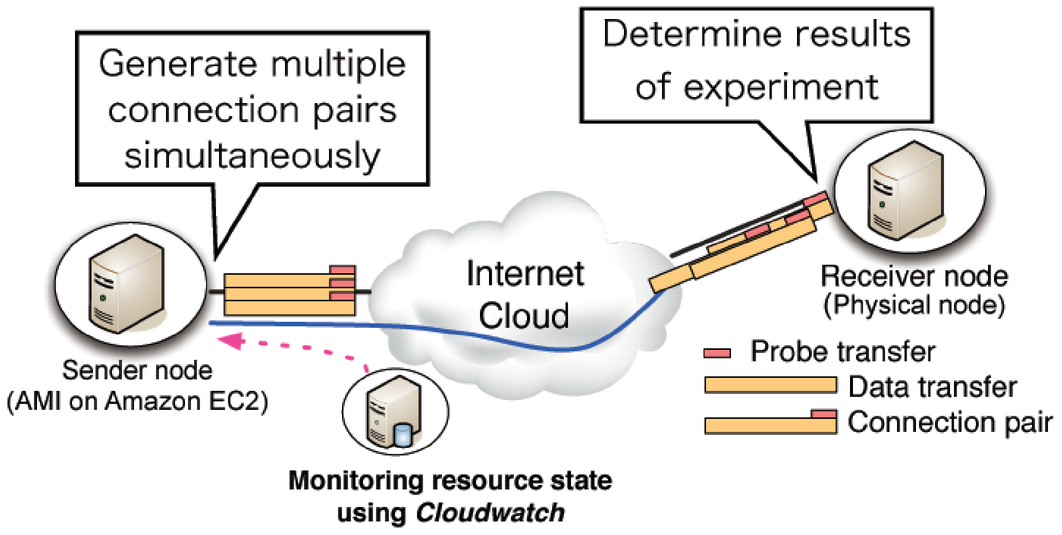

Generate multiple connection pairs simultaneously
Determine results of experiment
Internet Cloud
Receiver node (Physical node)
Sender node (AMI on Amazon EC2)
Probe transfer
Data transfer
Connection pair
Monitoring resource state using *Cloudwatch*

FIGURE 2.17: Throughput measurement method on Amazon EC2.

2.8.5 Throughput Measurement Results

Although we measured the throughput for the connection pair over 3 days, some of it was not gathered by libpcap error. A data set consisted of approximately 1,000 connection pairs over 18 hours. Throughput measurement results at indices *i* and *vi* are shown in Figure 2.18. Low network throughput was observed at all the throughput measurement results. While the results where the probe size was 16 KB at index *i* were clustered at the certain areas, the results where the probe size was 512 KB at index *vi* were widely changed with non-linear characteristics. The results from index *ii* to *v* were similar to the results at index *i*, and the others were similar to the results at index *vi*. The large-sized probe would be appropriate for the connection pair.

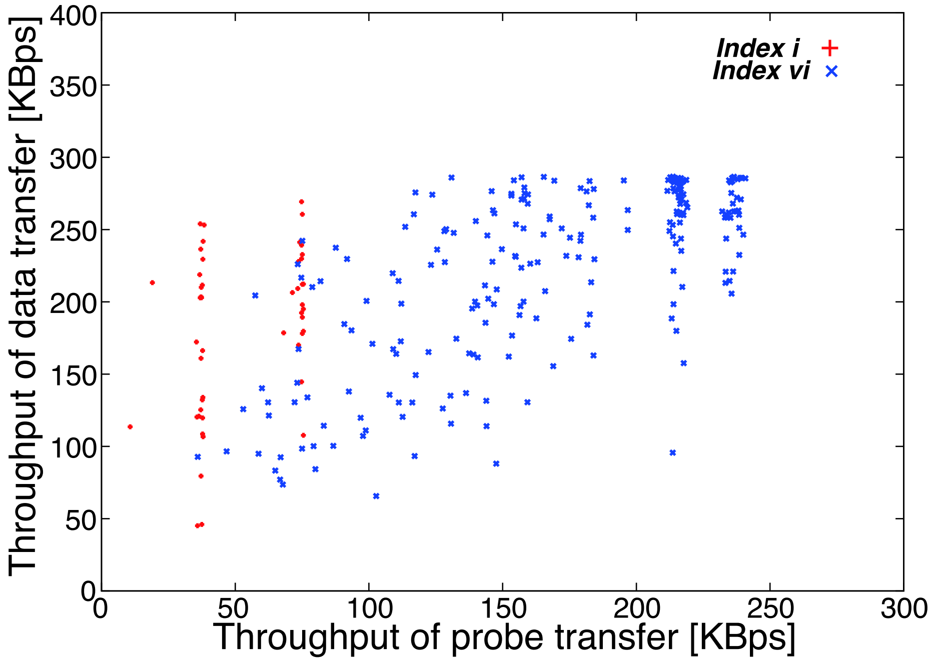


FIGURE 2.18: Connection pair throughput at indices *i* and *vi*.

2.8.6 Analysis of Network Metrics

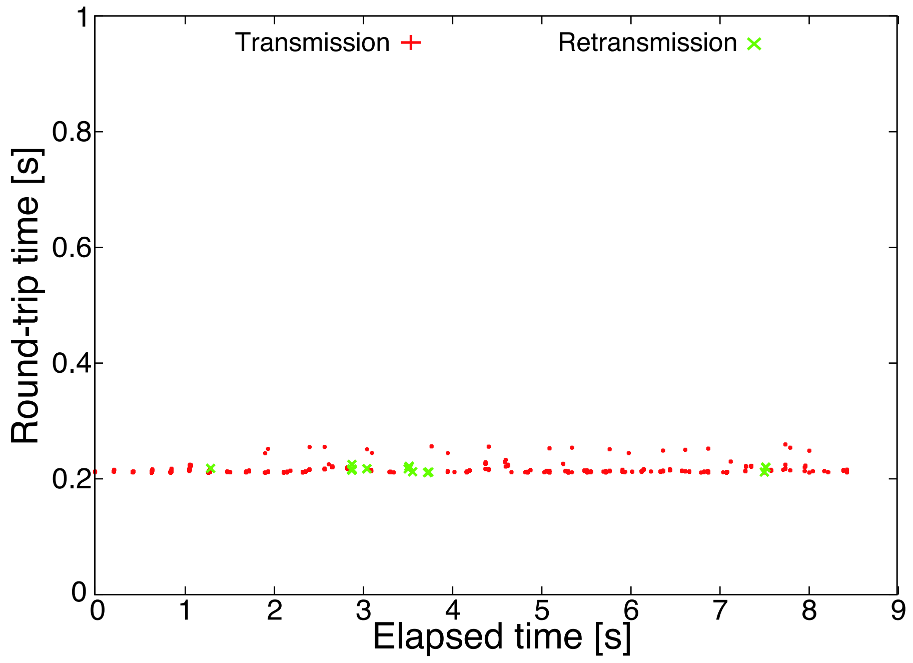
Here, we first analyze the network metrics to understand the traffic characteristics on Amazon EC2. Next, we observe the packet spacings for reproduction of the anomalies. Finally, we compare them to the results with the anomalous case on PlanetLab.

2.8.6.1 RTT and Packet Loss

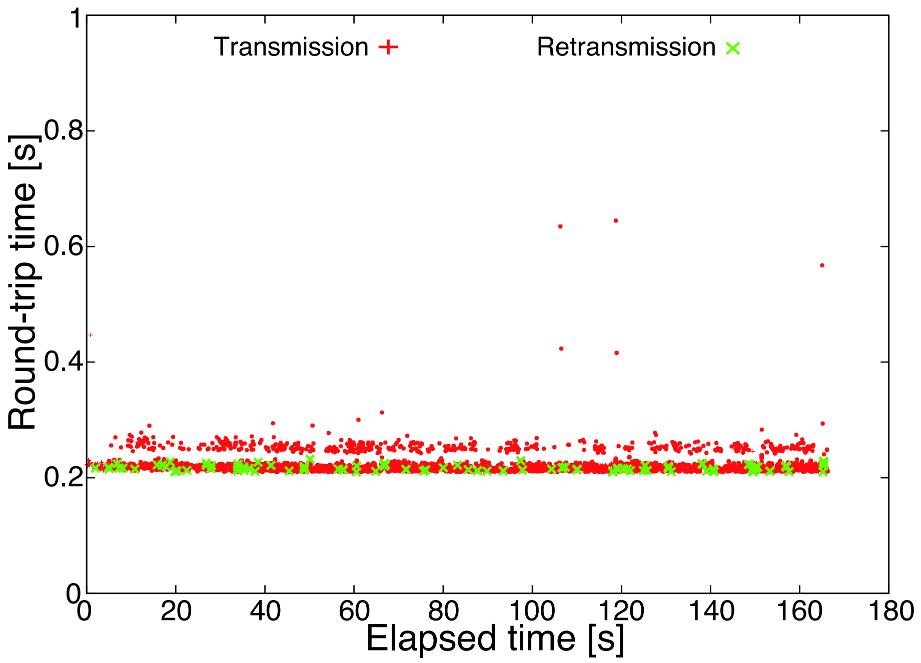
The RTT normally fluctuates and the network throughput decreases as a result of network congestion on the end-to-end path. Thus, a lot of packets would be dropped in the congestion. These well-known network metrics are important to understand Internet traffic characteristics. Figure 2.19 shows RTT of connection pair at index *ix*. The throughput measurement results of probe and data were 121.4 KBps and 98.5 KBps respectively. Although the throughput decreases were occurred, the most values of RTT were stable and the mean values of RTT of them were 0.216 [s] and 0.220 [s]. The values are similar to the mean RTT (0.213 [s]) using ping. Moreover, packet loss rate of the data transfer was 1.304 %. It would be small but the packet loss was continuously occurred during the throughput measurements. Table 6.2 shows mean values of the network metrics for all the indices. There were no significant changes in the mean RTT, so the network condition did not exhibit any abrupt changes. The mean values at all the indices were higher than that using ping because the sender node had to wait for a timeout. Although the mean values of RTT were not significantly changed, there was packet loss continuously. The loss rate at all the indices was less than 10 %, however the packet loss was similar to the above results.

TABLE 2.7: Mean values of well-known network metrics.

Index	**Mean RTT [s]** (Probe)	**Mean RTT [s]** (Data)	**Mean loss rate [%]** (Probe)	**Mean loss rate [%]** (Data)
i	0.215	0.218	2.025	1.675
ii	0.215	0.217	1.711	1.436
iii	0.215	0.217	1.86	1.462
iv	0.217	0.218	0.056	1.192
v	0.217	0.217	5.397	1.234
vi	0.216	0.215	1.122	0.581
vii	0.218	0.217	1.646	0.9
viii	0.218	0.218	1.775	0.794
ix	0.216	0.215	0.645	0.42
x	0.217	0.216	1.167	0.709



(a) Probe transfer (1 MB)



(b) Data transfer (16 MB)

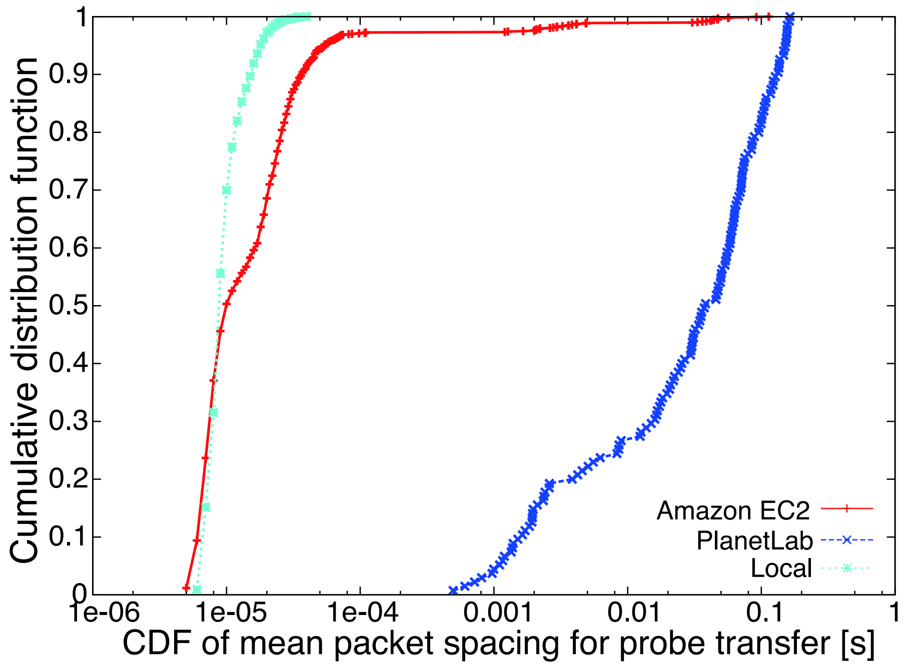
FIGURE 2.19: RTT of connection pair at index *ix*.

2.8.6.2 Packet Spacing

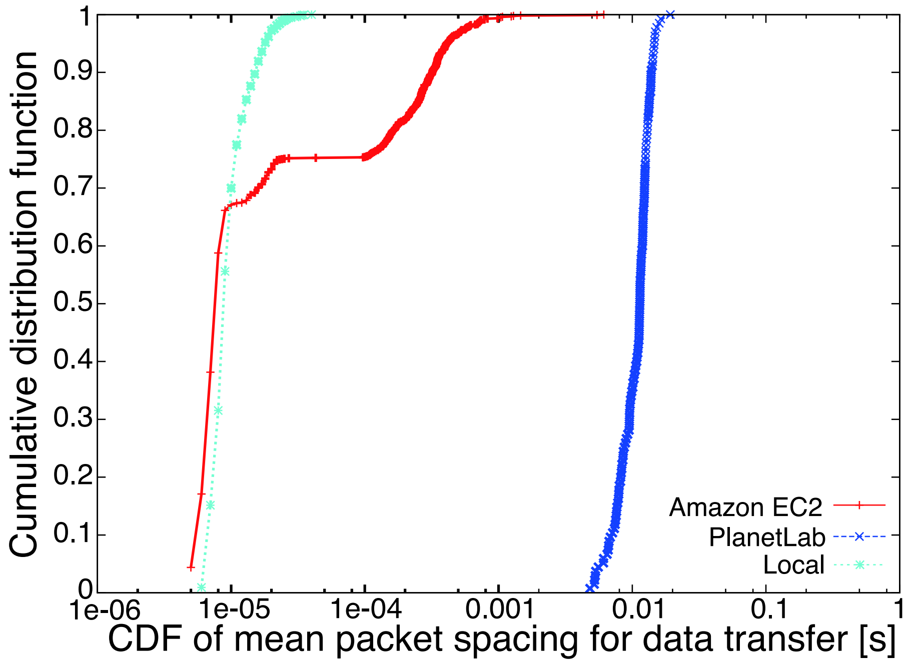
In order to observe the anomalies on Amazon EC2, we investigate the packet spacings for the connection pair. We also compare them to the results, where the probe size was 64 KB and the data size was 32 MB, which experience throughput instability caused by the anomalies on PlanetLab. Figure 2.20 shows the CDF of packet spacing on PlanetLab and the CDF of mean packet spacing for the connection pair at all the indices on Amazon EC2. Additionally, we insert the CDF of packet spacing on the local environment in Figure 2.8. The CDF of packet spacing on PlanetLab was larger than the others, and their pattern was the anomalous case. A part of CDF of packet spacing on Amazon EC2 was stable like the result on local environment. However, some of it was increased, and similar to the results on PlanetLab.

To identify the reproduction of the anomalies, we present that the case where the mean packet spacings of probe and data were 0.003620 [s] and 0.000415 [s] at index *ix*. Figure 2.21 shows the packet spacing at the above case. The throughput measurement results of probe and data were 193.9 KBps and 251 KBps respectively. Due to the values where packet spacings of probe and data were 0.469552 [s] and 0.475770 [s], the mean packet spacing was increased. In the other results, very few of packet spacings were the anomalies, and it is hard to determine the major cause of throughput instability. Our results are different from the results shown by Wang et al. [50]. Typically, the fluctuation in packet spacing can be caused by flow control in TCP mechanism. Packet will be not sent immediately after ACK packet. We observe advertised window to identify the effect of flow control. Figure 2.22 shows packet spacing and advertised window at the above case. The window is saturated to up maximum size, thus there is no impact of flow control.

To summarize the Internet traffic characteristics on Amazon EC2, very few of packet spacings were the anomalies and they were not the major cause of throughput instability. Packets were continuously dropped during the throughput measurements although RTT was very similar to mean RTT using ping. The packet loss was the major cause of throughput instability. In the data centers, tens of nodes per rack are connected via top of rack switches that connect to high degree aggregation switches [53]. The previous works [53][54] presented impairments among flows in the data center. Our implications of throughput instability are that various types of many flows co-exist in the data center and available buffer size in switches would be insufficient. These would affect the measurement results.

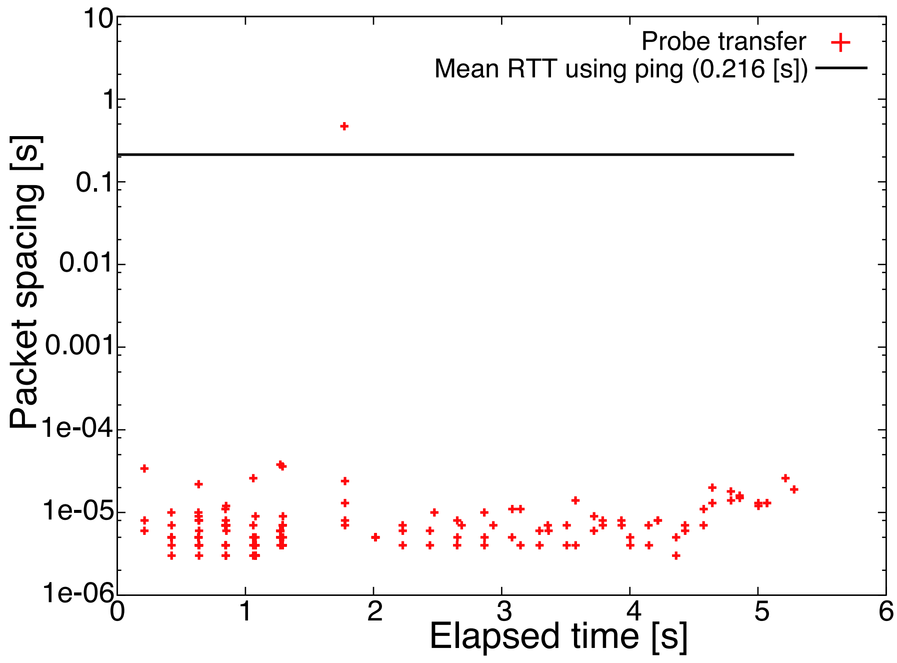


(a) Probe transfer

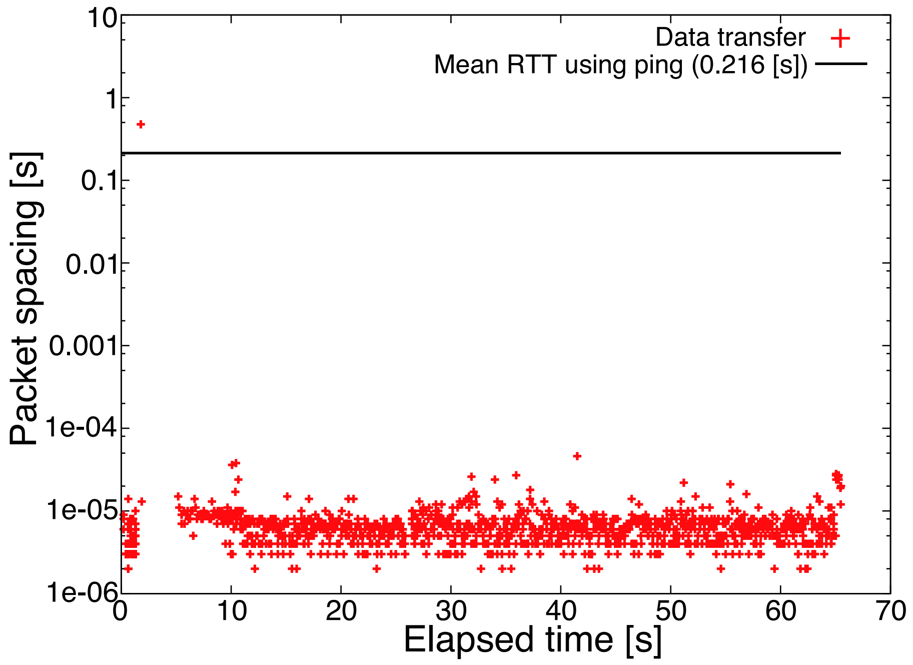


(b) Data transfer

FIGURE 2.20: CDF of mean packet spacing at all indices on Amazon EC2, PlanetLab, and local environment.

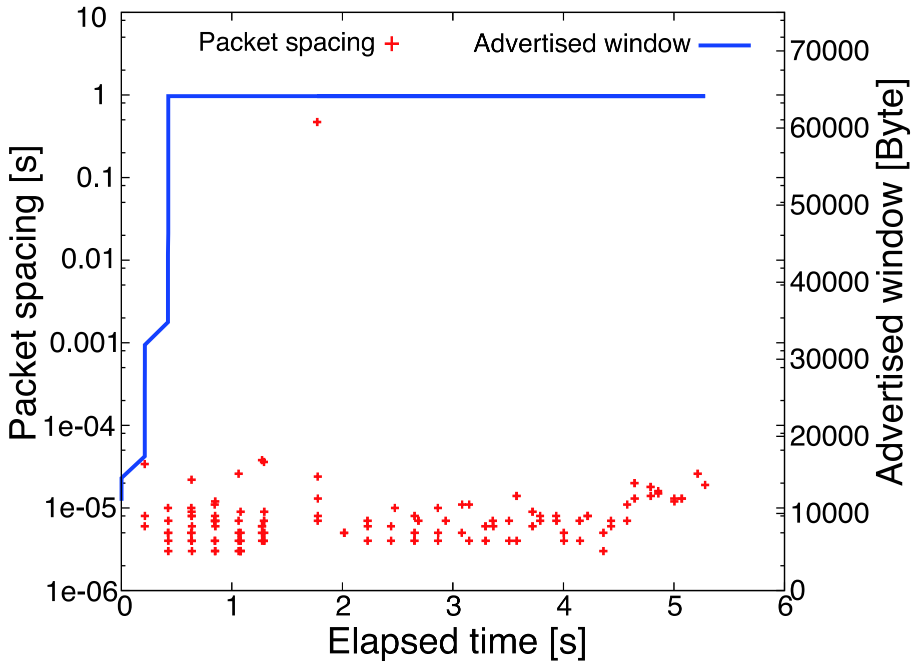


(a) Probe transfer (1 MB)

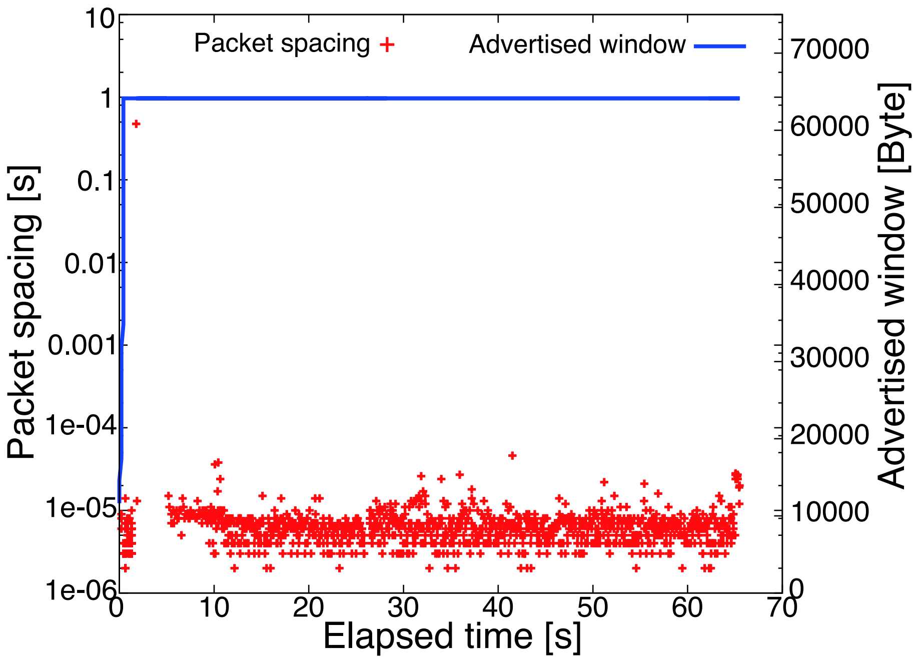


(b) Data transfer (16 MB)

FIGURE 2.21: Packet spacing of outgoing transfers at index *ix*.



(a) Probe transfer (1 MB)



(b) Data transfer (16 MB)

FIGURE 2.22: Packet spacing and advertised window of outgoing transfers at *ix*.

2.8.7 CPU Utilization

Because the packet spacings are similar to those over the non-virtualized environment, the CPU utilization will be low. Figure 2.23 shows the CPU utilization at indices *i* and *vi*. Most of values are less than 20 %. In the experiments, the CPUs were not busy, and the CPUs were fully allocated to the instances. The others were similar to the above cases. In cloud data centers, the CPU utilization would be low due to resource policy and the structure of data center. Again, very few of anomalies were occurred on Amazon EC2, and they were not the major cause of throughput instability.

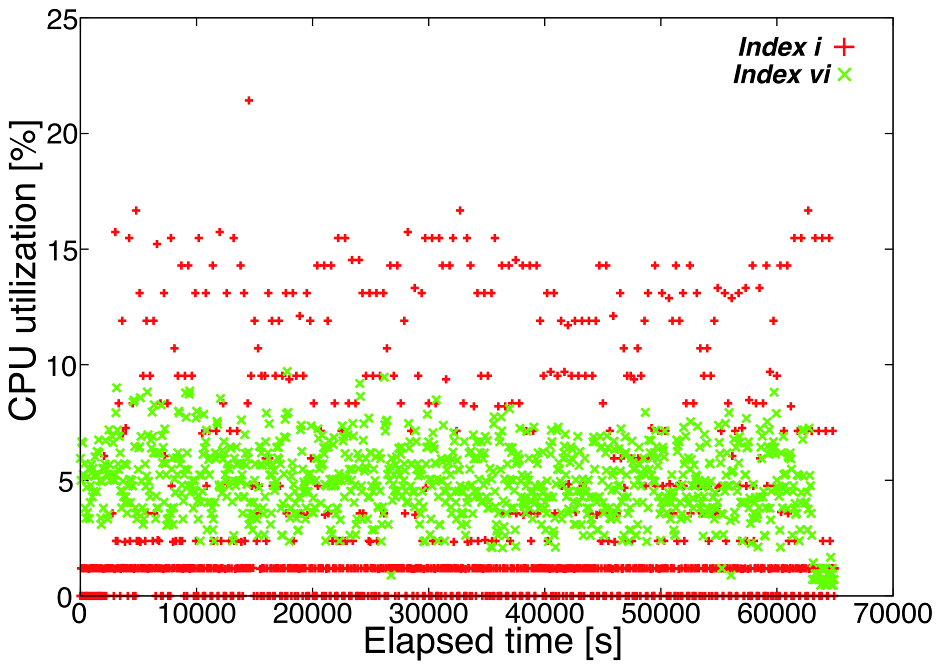


FIGURE 2.23: CPU utilization on Amazon EC2 (indices *i* and *vi*).

2.9 Conclusion

This chapter shows the importance of packet spacing for the prediction of network throughput in virtualized network environment, and presents a naive method for estimating the negative impact of virtualization. Again, it is important to understand the traffic characteristics for a precise prediction model. Research issues, which address precise network measurement on virtualized environments, were presented and previous works, which can help to diagnose the research problems and to find alternative solutions, have also been discussed. In conclusion, there is the throughput instability to disturb precise network measurement when the virtualization technology is used. The major cause is the oversize packet spacing occurred by CPU scheduling latency among virtual platforms. If the network throughput is decreased by the oversize packet spacings, we should carefully review measurement results. Furthermore, solutions that estimate the negative impact on userland have not been addressed in the existing literature and there are no adequate solutions that focus on empirical-statistical criteria. Consequently, this chapter has established the adequate solution of this thesis's research problems described in Chapter 1. The Internet traffic characteristics and the naive estimation method have been published in [20][21]. The Internet traffic characteristics on Amazon EC2 have been also published in [22].

The following chapter discusses the problem of naive method. In order to overcome the problem, we propose an alternative estimation method using principal component analysis (PCA). In this chapter, we also present the evaluation results of alternative estimation method.

Chapter 3

Estimation of Traffic Anomaly on the Internet

3.1 Overview

In this chapter, we mainly focus on a problem of naive method and an alternative estimation method to overcome the problem. The remainder of this chapter is divided into six more sections. Section 3.2 describes the problem of naive method. Section 3.3 introduces some studies related to monitoring systems on the virtualized testbed. In this section, we diagnose their characteristics and discuss their missing points. Next, we show throughput measurement method with resource state and measurement results in Section 3.4. Then, our method for estimating the anomalous case and its validation are presented in Sections 3.5 and 3.6. Finally, we conclude with a summary of the main points in Section 3.7.

3.2 Problem of Naive Method

Although the CPU availability is an important criterion for estimating the anomalous case, a naive method for measuring it may overconsume CPU resources and can affect the performance of other tasks. Thus, the anomalous cases would be occurred by our monitoring program when there are a few of CPU resources. It is important to overcome this problem for gathering precise learning samples. The modification of kernel is a possible way to overcome the problem, but is would be inappropriate to the virtualized network testbed. We focus on a nature of resource state through statistical analysis to find an appropriate estimation method.

3.3 Monitoring Systems

We here introduce some monitoring systems on the virtualized network testbed.

CoMon [55][56] is a centralized resource monitoring system for the virtualized testbed. It provides views of the virtualized testbed, such as node-centric and slice-centric information. Moreover, it has been used for selecting nodes and identifying problems on the virtualized testbed. Because it gathers data every five minutes, the data granularity is limited and the data type makes it hard to estimate fluctuation in packet spacing.

Clue [57] is an anomaly detection system for the virtualized testbed. However, this system focused on detecting anomalous behavior for the virtualized testbed and it used data on CoMon only.

Slicestat [43] provides slice-level resource consumption information, such as CPU, memory utilization, network I/O, number of processes, and so on, at each node on the virtualized testbed. It does not provide node-level information, such as SSH failing and shutdown. In these monitoring systems, however, the authors did not discuss any anomalous cases occurring in their network experiments.

3.4 Throughput Measurement with Resource State

In this section, we first introduce how to compose node pairs for the throughput measurement with resource state. Next, we investigate the anomalous case in packet-level analysis. Finally, we show features of resource state when the anomalous cases were and were not present.

3.4.1 Node Characteristics

To measure the throughput with the resource state, we empirically selected seven pairs of nodes (fourteen nodes), which we refer to these pairs as (α, β), (γ, δ), (ϵ, ζ), (η, θ), (κ, λ), (μ, ν), and (ξ, π). These nodes are located at different sites across North America and Europe, and are composed of one, two, four, or eight independent CPU cores, and have physical memory of approximately 3 GB, excepting node λ. We measured hop count using traceroute and RTT using ping; their basic node characteristics are given in Table 3.1.

We generated various sizes of connection pairs to observe the anomalous case and to establish a throughput prediction method for network virtualization. We observed the resource state per slice on the node with slicestat every 1 minute during throughput measurement. The packet spacing is hidden to users, and we cannot measure it simply. Our approach of observing the resource state on the node is suitable for anomaly estimation. The measurement methodology is the same to that in Figure 2.4 and the sizes of the connection pairs are also the same to those in Table 2.2.

TABLE 3.1: Node characteristics on virtualized testbed.

Node name	**CPU speed [Ghz]**	**CPU cores**	**Memory [GB]**	**Node pair** (arrow denotes transfer direction)	**Mean RTT [ms]**	**Hop Count**
α	Xeon 2.5	4	3.29	$\alpha \leftarrow \beta$	38.4	18
β	Pentium4 3.2	2	3.03			
γ	Core2Quad 2.66	4	3.42	$\gamma \leftarrow \delta$	20	15
δ	PentiumD 3.2	2	3.47			
ϵ	Xeon 2.4	4	3.53	$\epsilon \leftarrow \zeta$	49.9	19
ζ	PentiumD 3.2	2	3.54			
η	Core2Quad 2.66	4	3.42	$\eta \leftarrow \theta$	57.6	15
θ	Core2Quad 2.66	4	3.21			
κ	Xeon 2.66	8	3.28	$\kappa \leftarrow \lambda$	43.7	17
λ	Athlon(tm) 64 3200+	1	2.01			
μ	Xeon 2.4	2	3.03	$\mu \leftarrow \nu$	28.3	14
ν	PentiumD 3.2	2	3.54			
ξ	Xeon 2.66	2	2.96	$\xi \leftarrow \pi$	42.8	17
π	Xeon 2.4	4	3.46			

3.4.2 Throughput Measurement Results

We measured the throughput for approximately 1,100 connection pairs from all the node pairs over 48 hours. Throughput measurement results are shown in Figure 3.1. Low network throughput was observed on two pairs (α, β) and (γ, δ). However, there were no significant changes in the RTT and packet loss rate on network paths between the sender and the receiver of the pairs, as shown in Table 3.2. Although the network metrics of the path from ζ to ϵ were smaller than those of the path from θ to η, the throughput from ζ to ϵ was low.

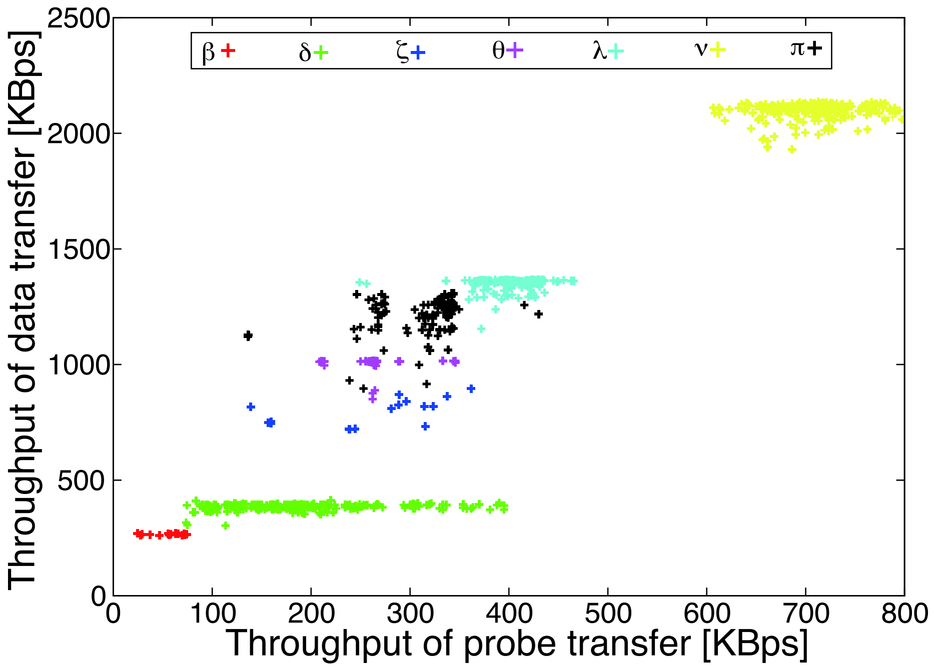


FIGURE 3.1: Actual throughput for connection pair at all pairs.

TABLE 3.2: Mean RTT and packet loss rate of connection pair.

Node pair	RTT [ms]	Loss rate [%]
α - β	40.5	0.0
γ - δ	22.1	0.002
ϵ - ζ	52.2	0.0
η - θ	60.2	0.478
κ - λ	44.9	0.001
μ - ν	29.2	0.075
ξ - π	48.5	0.069

3.4.3 Cause of Throughput Instability

To clarify the cause of throughput instability, the mean packet spacing and the throughput of the connection pairs at all the sender nodes is shown in Figure 3.2. The mean packet spacing at node pairs (α, β), (γ, δ), and (ϵ, ζ), which experienced low throughput, was larger than that of the other node pairs. Thus, it can be concluded that the anomalous cases were the cause of throughput instability. Furthermore, the mean packet spacing and throughput decreases at node β were larger than those at the other nodes.

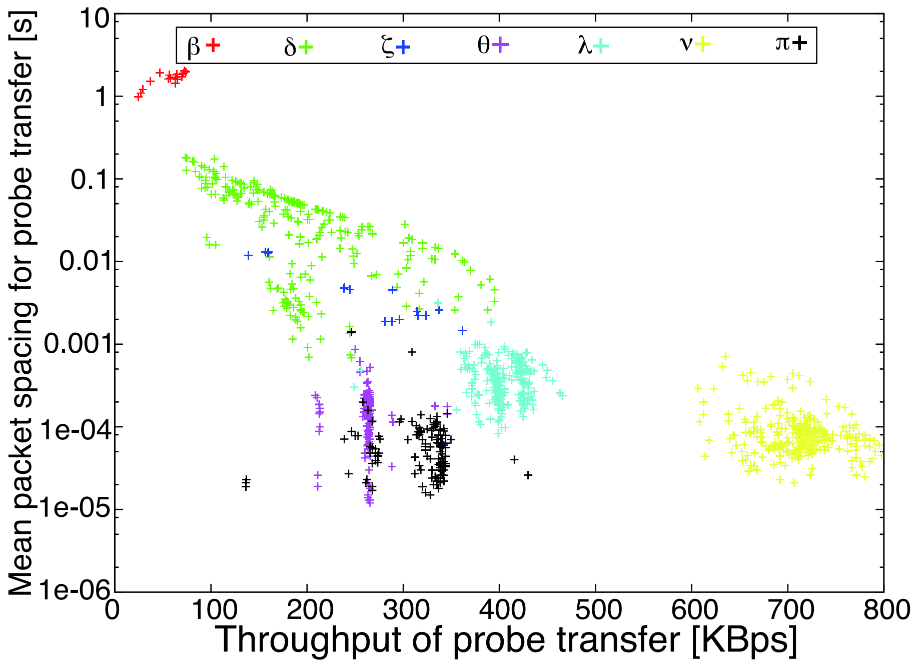
Then, the CDF of packet spacing at node δ is shown in Figure 3.3. The packet spacing was increased during the measurement, and some of packet spacing was larger than the mean RTT. Moreover, the forms of CDF for the connection pair are very similar to those of the anomalous cases in Figure 2.13. Similar results were also observed on nodes β and ζ.

To summarize, nodes β (red points), δ (green points), and ζ (blue points) experienced throughput instability caused by the anomalous case. We simultaneously measured both network throughput and the resource state at all the pairs to observe the characteristics of the resource state when the anomalous cases are present and to make a data set for a PCA. The data set consisted of approximately 8,000 resource state results.

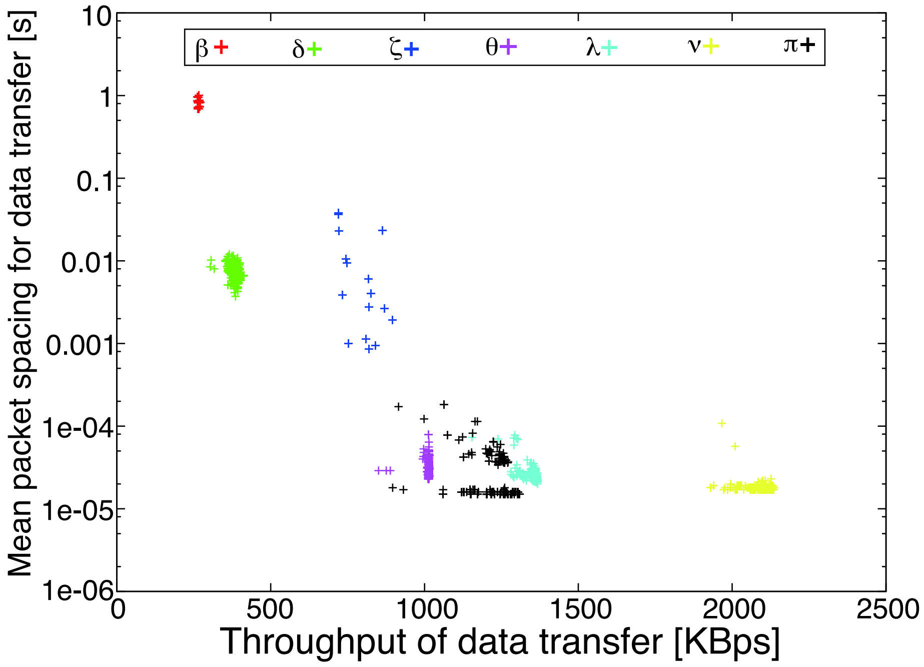
3.4.4 Features of Resource State

We here introduce criteria of resource state and their statistics. We excluded transmitting and receiving bandwidth from the resource state features; we included slice states because the anomalous cases relate to the resource state and multiple slivers simultaneously share the resources on the node. For a PCA analysis, we selected seven resource state features: total CPU utilization (CpuR), total memory utilization (MemR), total number of processes (Proc), number of current processes (CurProc), number of live slices (LiveC), number of active slices (ActiveC), and total number of slices (TotalC). These features were provided by the system on the virtualized network testbed. The CPU utilization and memory utilization were the consumption rates (%) at all the slices on the node. A process using the CPU cycle at the moment of measurement is called a *current process*. We call a slice that uses at least 0.1% of the CPU a *live slice*, and a slice that contains a process an *active slice*.

Mean usage of the resource state at all the sender nodes is shown in Table 3.3. There were large deviations in the usage at node θ, but no large deviation in the usage at node π. Additionally, the minimum, mean, and maximum CpuR at all the sender nodes is shown in Figure 3.4. Although the maximum CpuR at nodes λ, ν, and π rose to approximately 100%, the anomalous cases did not occur on these nodes. Thus, CpuR would be inappropriate for estimating the anomalous case. Moreover, it would be hard to design an anomaly estimator with all the features.



(a) Probe transfer



(b) Data transfer

FIGURE 3.2: Mean packet spacing and actual throughput at all sender nodes.

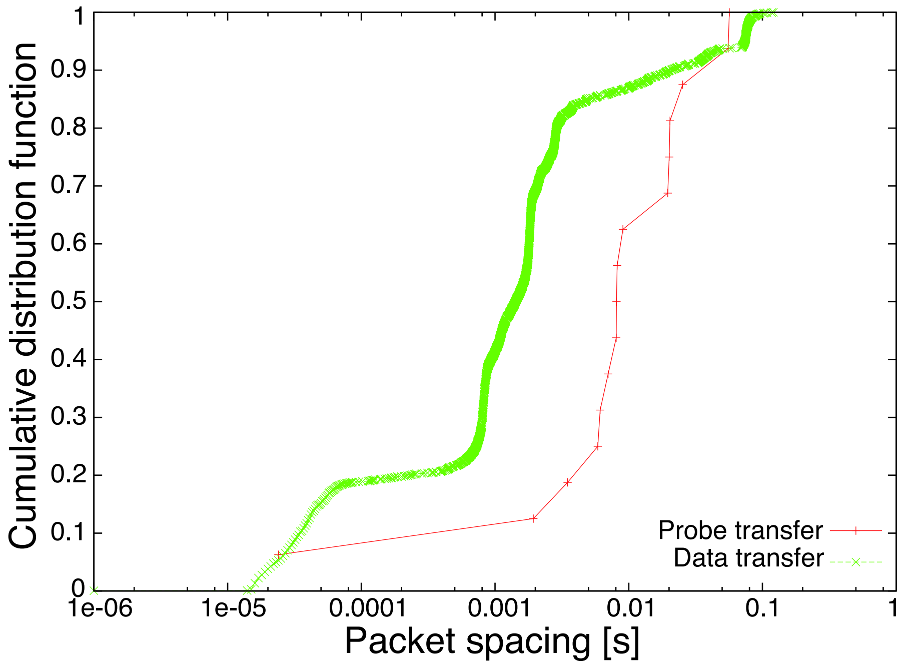


FIGURE 3.3: CDF of packet spacing at node δ.

TABLE 3.3: Mean usage of resource state at all sender nodes.

Sender node	**Anom-alies**	**CpuR [%]**	**MemR [%]**	**Proc**	**CurProc**	**LiveC**	**ActiveC**	**TotalC**
β	Yes	73.43	78.33	1021	13.33	12.71	136.3	149
δ	Yes	77.77	63.54	638.1	11.48	10.1	107.6	117.7
ζ	Yes	85.54	99.92	852.3	16.55	9.827	65.4	75.23
θ	No	21.65	29.52	335.4	1.617	1.606	37.5	39.11
λ	No	84.9	61.61	428.7	6.137	5.63	43.85	49.48
ν	No	91.05	47.95	639.7	5.244	5.085	76.99	82.07
π	No	62.61	55.39	687	8.833	6.408	149.8	156.2

3.5 A Method for Estimating Anomalous Case using PCA

In previous section, we showed the characteristics of throughput measurements and resource states. However, we cannot which resource state is important for estimating the anomalous case.

In this section, our estimation method using PCA is presented. As a first step, we introduce PCA and how to apply PCA into network research. Next, our analysis results using PCA are shown in detail. Finally, we describe boundaries to estimate the anomalous case automatically.

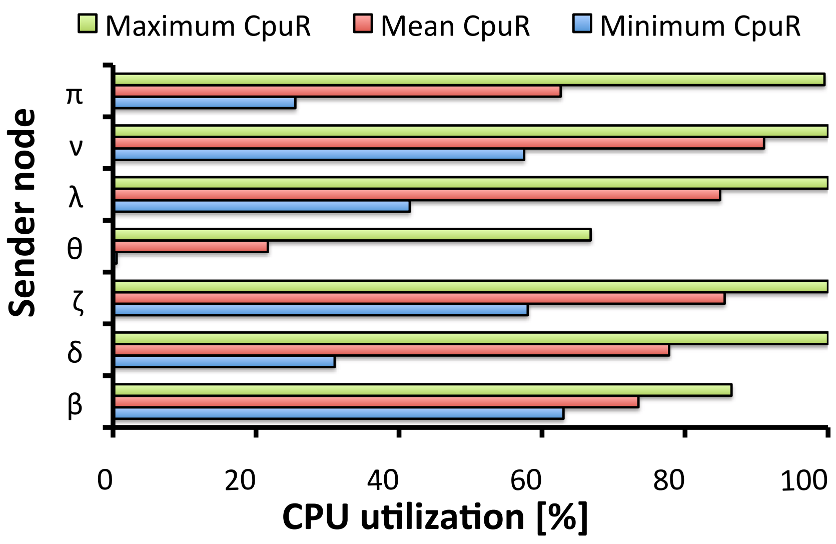


FIGURE 3.4: Minimum, mean, and maximum CpuR at all sender nodes.

3.5.1 Principal Component Analysis (PCA)

PCA is a dimensionality reduction technique that is widely used for applications such as lossy data compression, feature extraction, and data visualization. It has also been used in Internet traffic analysis for security [58][59]. It seeks a space of lower dimensionality, known as the principal subspace, such that the orthogonal projection of data points onto this subspace maximizes the variance of the projected points. An alternative definition of PCA is based on minimizing the sum-of-squares of the projection errors. Consider the general case of an M-dimensional projection space; the optimal linear projection for which the variance of the projected data is maximized is defined by the M eigenvectors $\boldsymbol{u}_1, \boldsymbol{u}_2, ..., \boldsymbol{u}_i, ..., \boldsymbol{u}_M$ of the data covariance matrix $\boldsymbol{S}$ corresponding to the largest M eigenvalues $\lambda_1, \lambda_2, ..., \lambda_i, ..., \lambda_M$. The eigenvector $\boldsymbol{u_1}$ is the first principal component, and it has the maximum entropy of give data set. The eigenvalues are arranged from largest to smallest, i.e., $\lambda_1 \geq \lambda_2 \geq, ..., \lambda_i \geq, ..., \geq \lambda_M$. Moreover, they are expressed as a percent of the total variance. The cumulative percentage of total variance s_i is accounted for by the current and all percentages. The sum of all the eigenvalues is equal to the number of variables. There are no obvious meaningful components from the trivial components. Most researchers would agree that the first and second components are probably meaningful, but it is difficult to decide on the exact meaningfulness. On the basis of earlier PCA studies [60][61], we adopted "$\lambda_i > 0.7$ or $s_i > 0.9$" as criteria for selecting meaningful components.

3.5.2 Analysis Result using PCA

We applied PCA to an approximately 7×8000 matrix of resource states. The eigenvalues and the cumulative percentage of total variance are described in Table 3.4. The top two principal components accounted for 84% of the cumulative percentage, and they are sufficient for describing the original resource state. Component loadings are the correlation coefficients between the variables and principal components: $Component\ loading = \sqrt{\lambda_i}\boldsymbol{u}_i$. These values are shown in Table 3.5.

TABLE 3.4: Eigenvalues and cumulative percentage of total variance.

PC	**1**	**2**	**3**	**4**	**5**	**6**	**7**
λ_i	4.844	1.043	0.651	0.313	0.126	0.023	0.000
s_i	0.692	0.842	0.934	0.983	0.995	1.000	1.000

TABLE 3.5: Component loadings.

PC	**1**	**2**	**3**	**4**	**5**	**6**	**7**
CpuR	0.299	-0.324	0.768	0.455	0.076	0.036	0.000
MemR	0.385	-0.438	-0.121	-0.239	-0.761	0.082	0.000
Proc	0.4	-0.213	0.11	-0.651	0.472	-0.365	0.000
CurProc	0.412	-0.186	-0.402	0.158	0.4	0.67	0.000
LiveC	0.401	0.043	-0.417	0.519	0.015	-0.617	-0.011
ActiveC	0.355	0.583	0.186	-0.123	-0.129	0.158	-0.063
TotalC	0.379	0.53	0.105	-0.033	-0.113	0.049	-0.073

As mentioned earlier, the first principal component captures the maximum entropy of given data set. In the analysis results, the first component has a positive correlation with all the resource features. It depicts workloads on the node. If the principal component score of the first component is close to the negative, the resources on the node are idle. Conversely, if the score of the first component is close to the positive, the resources are in busy states. For example, when resources, such as CPU and memory, are consumed by users, the other resource features are increased. The second component has a negative correlation with CpuR, MemR, Proc, and CurProc and a positive correlation with LiveC, ActiveC, and TotalC. The second component relates to the difference in the resource state. In the second component, MemR and ActiveC were larger than CpuR. This implies that these features are more important than CpuR and that an anomaly estimator can be designed without CpuR. The other components have very small eigenvalues, and it would be hard to describe the original resource state by using them.

We show a scatter plot of the first and second component scores in Figure 3.5. It provides a simple view of the resource state for anomaly estimation. There are seven clusters in the scatter plot, each of which represents the node used in the measurement. The Greek characters indicate the names of the sender nodes. Certain scores–such as -1, 0, 1, and so on–have no meaning in the scatter plot. Recall that nodes β, δ, and ζ are the nodes that experienced the anomaly. The clusters of the nodes situated on the bottom-right corner of Figure 3.5. This suggests that the combination of the higher score of the first component and the lower score of the second component might be used as an indicator of anomaly. The scores of the first component at clusters λ, ν, and θ are in the negatives. These correspond to the state at nodes λ, ν, and θ showing that there are no heavy workloads on these nodes.

To summarize, the first component can be interpreted as workloads and the second one as lack of resources, leading to the anomalous case. Thus, although the resources are busy, there are no anomalous cases when the resources are sufficiently allocated to users, i.e., at cluster π. Conversely, the anomalous case occurred when the resources were busy and there were not enough of them for allocation, i.e., at clusters β, δ, and ζ. Thus, the first and second components can be used instead of CPU availability to estimate the anomalous case.

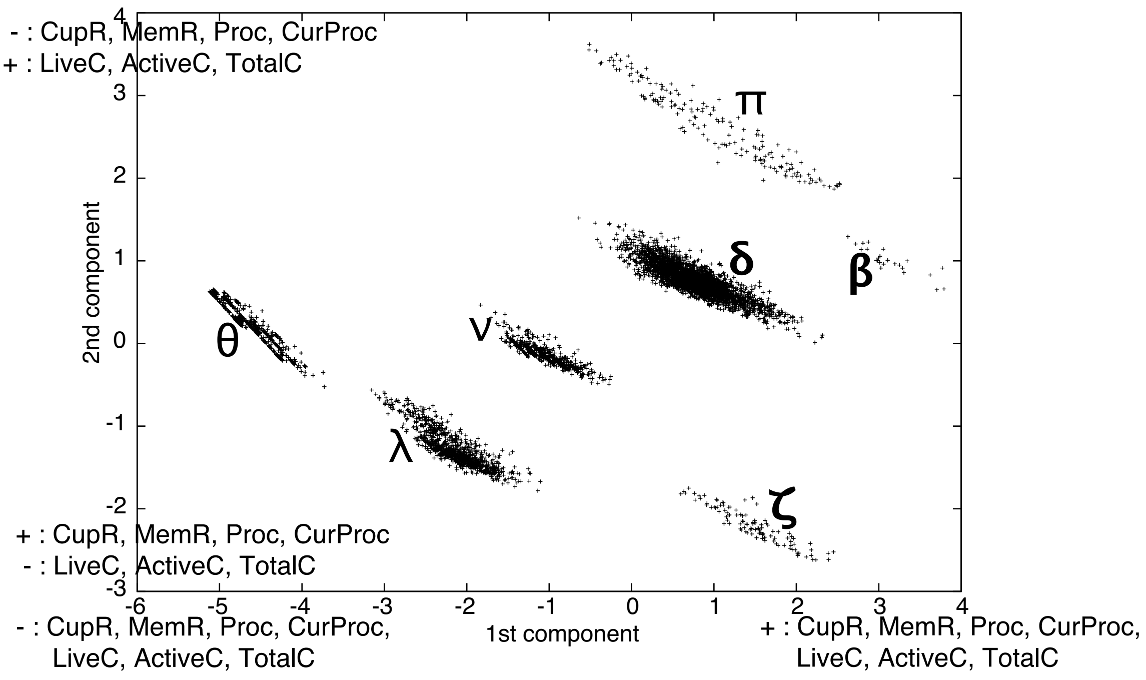


FIGURE 3.5: First and second component scores.

3.5.3 Boundaries for Estimating Anomalous Case

In the previous section, we showed that the resource state of anomalous case is clustered at certain areas in the space spanned by the first and second eigenvectors. To estimate the anomalous case automatically, we need to determine the boundaries of the normal and anomalous cases. We used Bayes' approach to find appropriate boundary values for the component scores. The analysis results of PCA were used, and the minimum $error(e)$ on the component scores was found. This error is defined as

$$\begin{aligned} error(e) &= P\left(A|PC_1 < x, PC_2 > y\right) \\ &\quad + P\left(O|PC_1 > x, PC_2 < y\right) \end{aligned} \tag{3.1}$$

where A and O are the anomalous case (β, δ, and ζ) and the normal case (λ, ν, θ, and π), respectively, PC_1 is the first component scores, and PC_2 is the second component scores. $P(A|PC_1 < x, PC_2 > y)$ is the conditional probability of A, given $PC_1 < x, PC_2 > y$, and $P(O|PC_1 > x, PC_2 < y)$ is the conditional probability of O, given $PC_1 > x, PC_2 < y$. Moreover, $error(e)$ can be transformed by Bayes' theorem to

$$\begin{aligned} error(e) &= \frac{P(PC_1 < x, PC_2 > y|A)P(A)}{P(PC_1 < x, PC_2 > y)} \\ &\quad + \frac{P(PC_1 > x, PC_2 < y|O)P(O)}{P(PC_1 > x, PC_2 < y)} \end{aligned} \tag{3.2}$$

where $P(A)$ is the prior probability of the anomalous case and $P(O)$ is the prior probability of the normal case. In the learning phase, we investigated the values of x ranging from -1 to 0 and the values of y ranging from 1 to 2. When the value was smaller than 0.01, there was no change in the error. We increased both values by 0.01. We found the minimum error value (0.000299902) where the value of x was -0.26 and the value of y was 1.52. The boundaries are described in Figure 3.6. Moreover, we evaluated Type I (false positive) and Type II (false negative) errors of the normal and anomalous cases using the boundaries. The Type I error occurred in 0.022% and the Type II error in 0.210%. Although the Type II error is larger than the Type I error, it is still very small. Our approach is suitable for finding appropriate values because we analyzed the meanings of all the clusters and found the anomalous cases to be clustered in a certain area, i.e., the clusters of nodes β, δ, and ζ in the bottom-right corner of Figure 3.5. If the first component score is greater than -0.26 and the second component score is less than 1.52, an anomaly can be assumed.

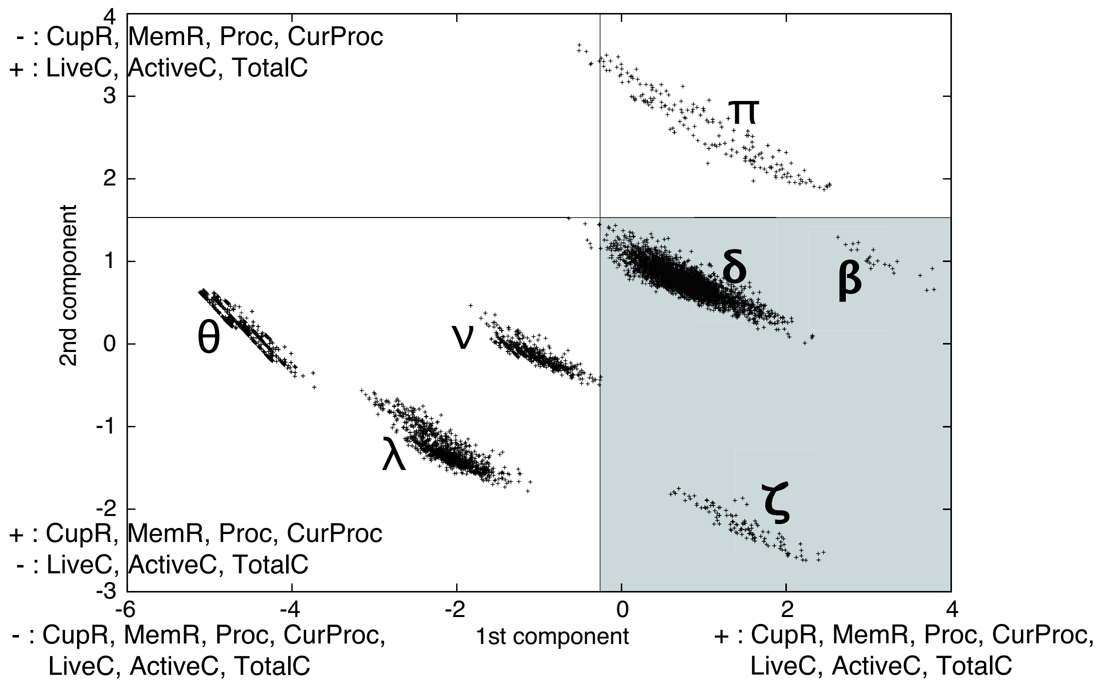


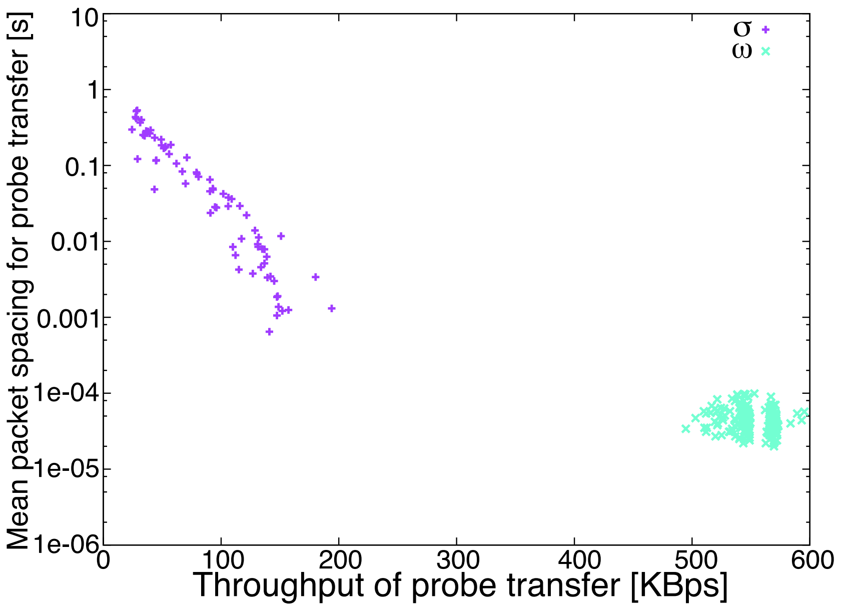
FIGURE 3.6: Component scores with boundaries.

3.6 Evaluation Results using PCA

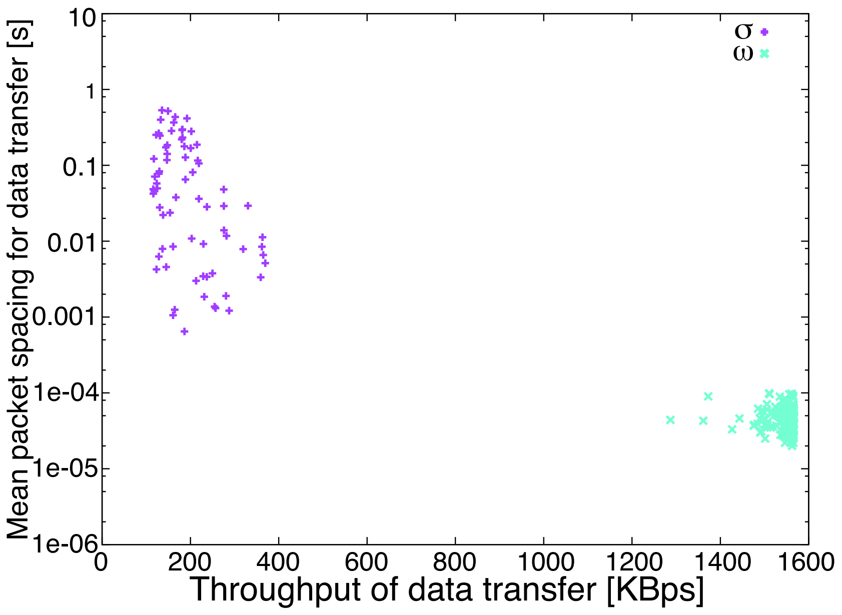
In this section, we focus on the validation of our estimation method. We present the composition of input data set to evaluate our method. Here, the validation of our method is presented with the input data set.

3.6.1 Input Data Set

In order to show the validation of our estimation method, we additionally selected two pairs of nodes (four nodes), which we refer to the pairs as (ρ, σ) and (ψ, ω). These pairs are located at different sites across North America. Their basic characteristics are given in Table 3.6. We measured the network throughput through the connection pair and observed the resource state with slicestat over 24 hours. We found the anomalous case at node σ. These are a major cause of throughput decreases. Mean packet spacing and throughput of the connection pair are depicted in Figure 3.7, and the mean usage of the resource state at nodes σ and ω are shown in Table 3.7. The input data set consisted of approximately 1,350 resource state results at nodes σ and ω, so the input data set was also aggregated at cases with and without the anomalous cases.



(a) Probe transfer



(b) Data transfer

FIGURE 3.7: Mean packet spacing and actual throughput at nodes σ and ω.

TABLE 3.6: Node characteristics of input data set on virtualized testbed.

Node name	**CPU speed [Ghz]**	**CPU cores**	**Memory [GB]**	**Node pair** (arrow denotes transfer direction)	**Mean RTT [ms]**	**Hop Count**
ρ	CoreDuo 2.33	2	3.45	$\rho \leftarrow \sigma$	90	21
σ	Xeon 3.4	2	2.97			
ψ	PentiumD 3.2	2	3.42	$\psi \leftarrow \omega$	39	14
ω	Core2Quad 2.66	4	3.5			

TABLE 3.7: Mean usage of resource state at input data set.

Sender node	**Anom-alies**	**CpuR [%]**	**MemR [%]**	**Proc**	**CurProc**	**LiveC**	**ActiveC**	**TotalC**
σ	Yes	73.58	80.1	662	15.8	14.54	102.3	116.9
ω	No	3.5	59.2	571	3	3	63.02	66.17

3.6.2 Evaluation Results

Here, we calculated the first and second component scores of the input data set. The scores at node σ were in the anomaly area. Conversely, the scores at node ω were not in the anomaly area. These scores are shown in Figure 3.8. The component scores show that there were heavy workloads at node σ and that there were not enough resources for allocation. However, the scores at node ω were not in the anomaly area because the workloads were not heavy. These results are sufficient to show the validity of our approach.

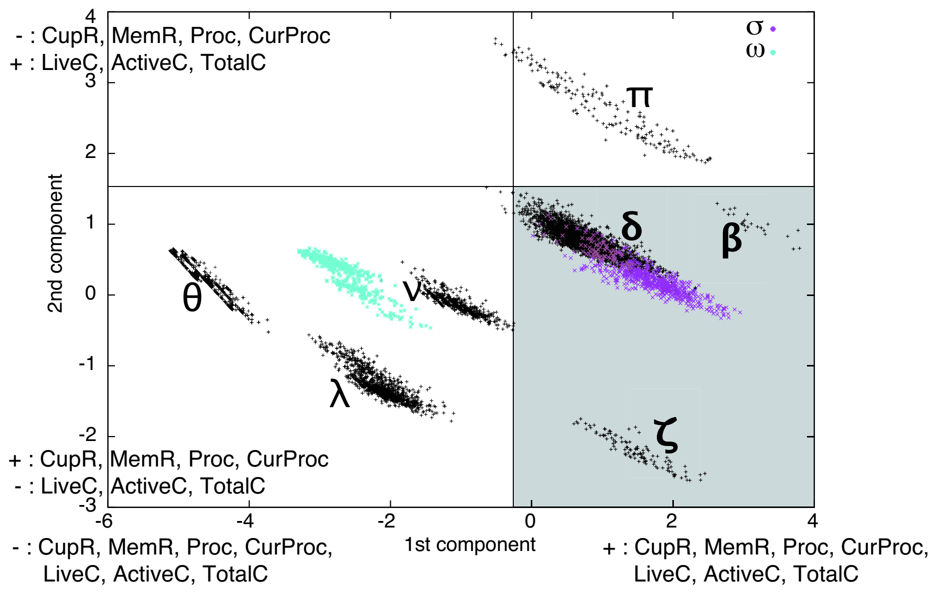


FIGURE 3.8: Component scores with boundaries and input data set.

3.7 Conclusion

This chapter presents the estimation method using PCA and the validation of estimation method. Our estimation method focuses on a nature of resource state with statistical approach, and it can prevent the overconsumed CPU resources. We applied PCA to a matrix gathered from the resource state to estimate the anomalous case. The analysis results showed that the top two principal components account for 84% of the original data set and can describe the original resource state. Component loadings and a scatter plot of the first and second component scores provided a simple but descriptive enough view of the resource state for the estimation. The first component can be interpreted as workloads, and the second one as lack of resources, leading to the anomalous case. We determined the appropriate boundaries of the components by using Bayes' approach and used them to automatically evaluate the anomalous case with an input data set gathered from other nodes. The evaluation results presented the validation of our approach. The estimation method using PCA is enough to answer research questions described in Chapter 1. The estimation method has been published in [23].

The following chapter identifies appropriate prediction parameters through statistic analysis. The parameters are identified in Chapter 4, and we propose a reasonable solution of the research questions. They will become the heart of our prediction method.

Chapter 4

Parameter Selection using Statistical Approach

4.1 Overview

The previous chapters introduced research issues of the Internet traffic characteristics when the virtualization technology is used. Contributions which have been achieved in our and previous works were also discussed in detail. It was ascertained that a few studies discussed the traffic characteristics on the virtualized network testbed and there are no adequate solutions to estimate the anomalous cases on userland. We proposed two estimation methods, which focus on resource state on userland. Again, it is very important to estimate the anomalous cases for measuring networks and gathering learning samples of prediction model.

In this chapter, our approach for the selection of an appropriate probe size is described before the prediction and the validity of the selected probe size is shown. We first introduce our assumption for the prediction model and a valid indicator for the selection in Section 4.2. Second, evaluation results of the connection pair are statistically shown through the indicator in Section 4.3. Next, we make clear how to fluctuate actual throughput for the connection pair and the appropriate probe size is selected through re-evaluation in Section 4.4. Then, the actual throughput with the selected probe size is shown in Section 4.5. Finally, we conclude with a summary of the main points in Section 4.6.

4.2 Proposed Approach for Parameter Selection

Various types of traffic, such as mice and elephant flows [62][63][64], co-exist in current networks and the elephant flow is a major cause of traffic spikes [62]. The probability distribution of traffic is close to long-tail, and it is too hard to determine a traffic model based on probability distribution. In this case, linear and parametric statistics methods would be inappropriate. In other words, non-parametric statistics methods are appropriate. These methods do not rely on a given probability distribution and focus on the ranks of observations in data sets. In particular, Spearman's rank correlation coefficient (ρ) assesses how well the relationship between data sets can be described using a monotonic function. We use ρ to denote the correlation coefficient of connection pair.

Here, we describe our assumption for the prediction model using the connection pair. We assumed monotonicity between probe throughput and data throughput for the prediction. For example, if probe throughput is decreased, data throughput will be decreased monotonically. Thus, we focused on the ranks of the connection pair, and Spearman's rank correlation coefficient (ρ) is a valid indicator for the selection. Moreover, a high ρ value implies that the probe will have a high predictability and that it can therefore be regarded as appropriate for prediction.

4.3 Evaluation Results for the Selection of Probe Size

In order to select an appropriate probe size for the precise prediction model using the connection pair, we first generate various sizes for the connection pairs every five minutes on the virtualized network testbed. Measurement methodology using the connection pair is described in Figure 4.1. Second, we gathered approximately 15,000 connection pairs from the virtualized testbed over thirteen days. We applied ρ to the data set to select an appropriate probe size for predicting the throughput. Sizes and their ρ are shown in Table 4.1. Again, a high ρ value implies that the probe will have a high predictability and it helps to build the precise prediction model. Although the scale and bandwidth of current networks are increasing, a 32-KB probe, which is smaller than the original probe size (64 KB), was the highest value of ρ (0.69). To clarify this result, we investigate the connection pairs in packet-level analysis.

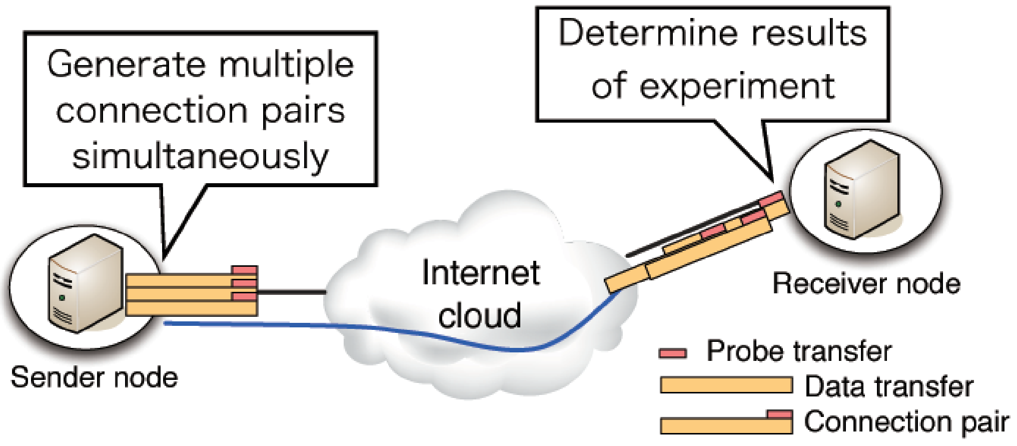

Generate multiple connection pairs simultaneously
Determine results of experiment
Internet cloud
Sender node
Receiver node
Probe transfer
Data transfer
Connection pair

FIGURE 4.1: Measurement methodology.

TABLE 4.1: Spearman's rank correlation coefficient for connection pairs.

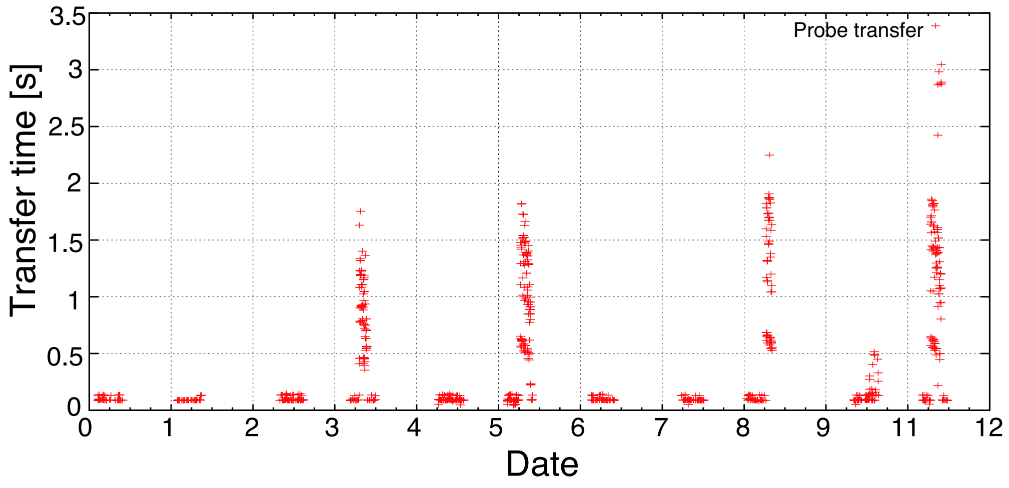
Index	**Probe size [KB]**	**Data size [MB]**	**Rank Cor. (ρ)**
i	16	8	0.49
ii	16	16	0.42
iii	32	16	0.69
iv	64	16	0.45
v	64	32	0.52
vi	128	64	0.42
vii	256	128	0.67
viii	256	256	0.45

4.4 Packet-level Analysis

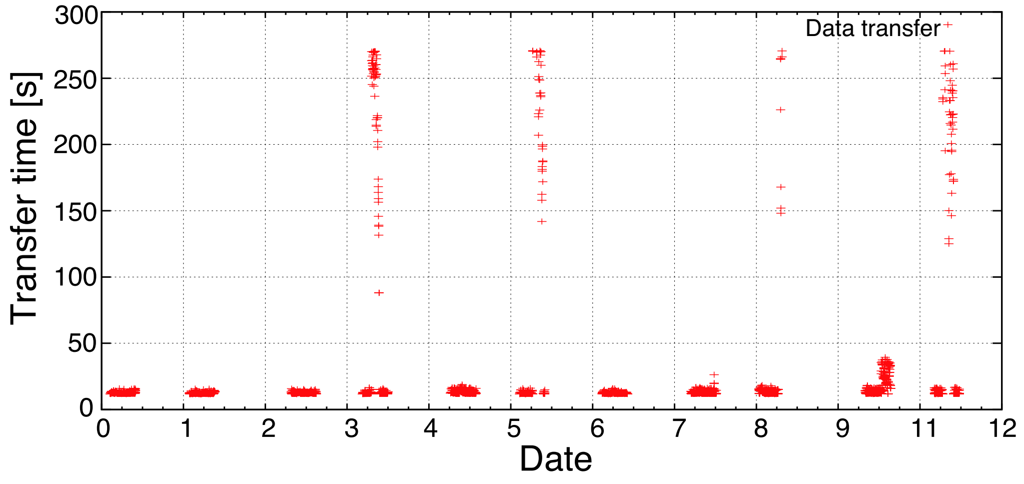
Here, we clarify the cause of throughput fluctuations in packet-level analysis. First, daily changes in the actual throughput are shown to observe throughput fluctuations. Next, we investigate the anomalous cases in the data set. Then, it will be clear how the anomalous cases can affect the actual throughput. Finally, we show the evaluation results without the anomalous cases, and select the appropriate probe size for our prediction model.

4.4.1 The Cause of Throughput Fluctuations

In order to investigate the cause of fluctuations, daily changes in the actual throughput are shown with the 32-KB probe and the 16-MB data. The actual throughput decreased in a certain time period. The daily changes in transfer time at index *iii* are shown in Figure 4.2. Two causes of throughput instability are considered: one is network congestion, and the other is the anomalous case. The throughput would be typically fluctuated by network congestion. This case can be predictable through the well-known network metrics. On the other hand, the throughput can be affected by the anomalous cases. Again, we explained and discussed them and their effects in Chapter 2.



(a) Daily change in transfer time at probe transfer (32 KB)



(b) Daily change in transfer time at data transfer (16 MB)

FIGURE 4.2: Daily change in transfer time at index *iii*.

In packet-level analysis, we found anomalous cases at index *iii*. To show the anomalous case at index *iii*, we introduce two cases (A and B). The respective mean RTT for the connection pair of the cases is 0.0456 [s] and 0.0443 [s] and respective packet loss rate is 0.098% and 0.188%. However, the actual throughput for probe transfer in the cases is 351.6 KBps and 18.8 KBps and that for data transfer is 1112.6 KBps and 234.3 KBps. The packet spacing in case A is vey short period, and it is similar to the non-virtualized environment. Case B experienced throughput drop caused by the anomalies, and the CDF of packet spacing in case B was different from that in case A. The CDF of packet spacing for connection pair at cases A and B is shown in Figure 4.3. The anomalous cases are found at both probe and data transfers. The mean packet spacing and throughput at index *iii* are described in Figure 4.4. The actual throughput is described when the mean packet spacing is increased. Moreover, we also found the anomalous cases from index *i* to *v*.

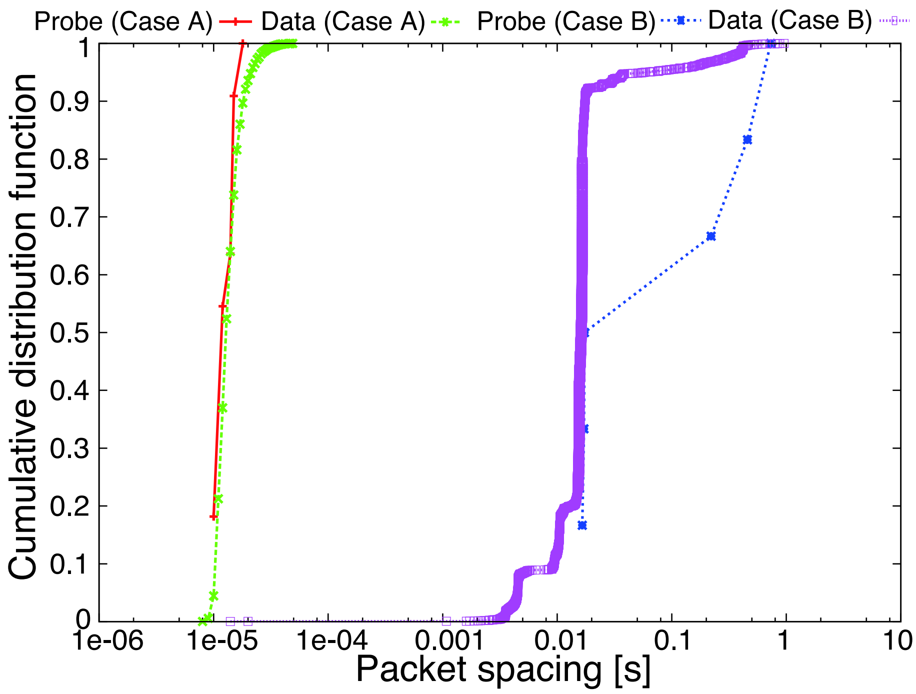
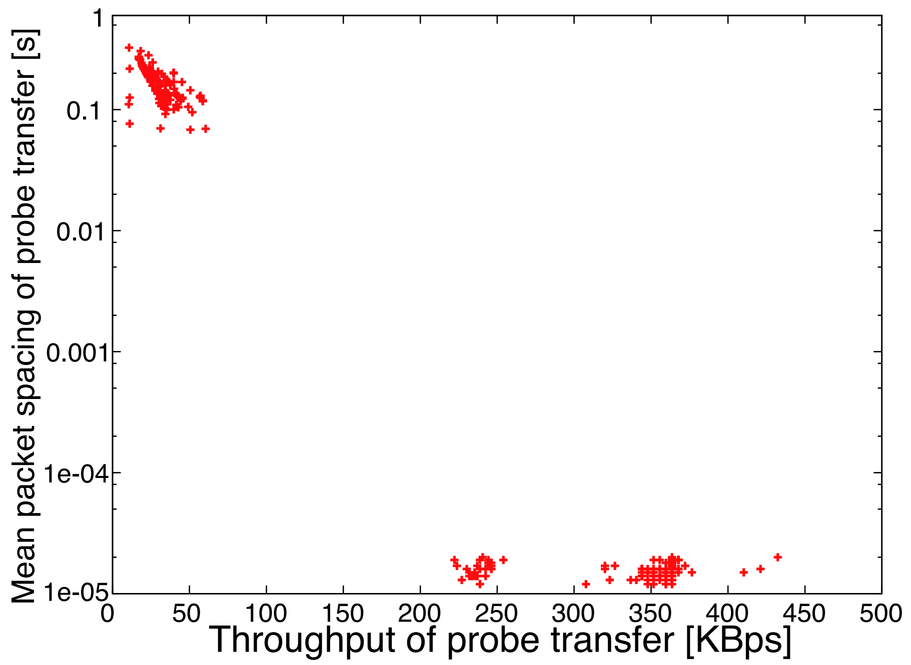
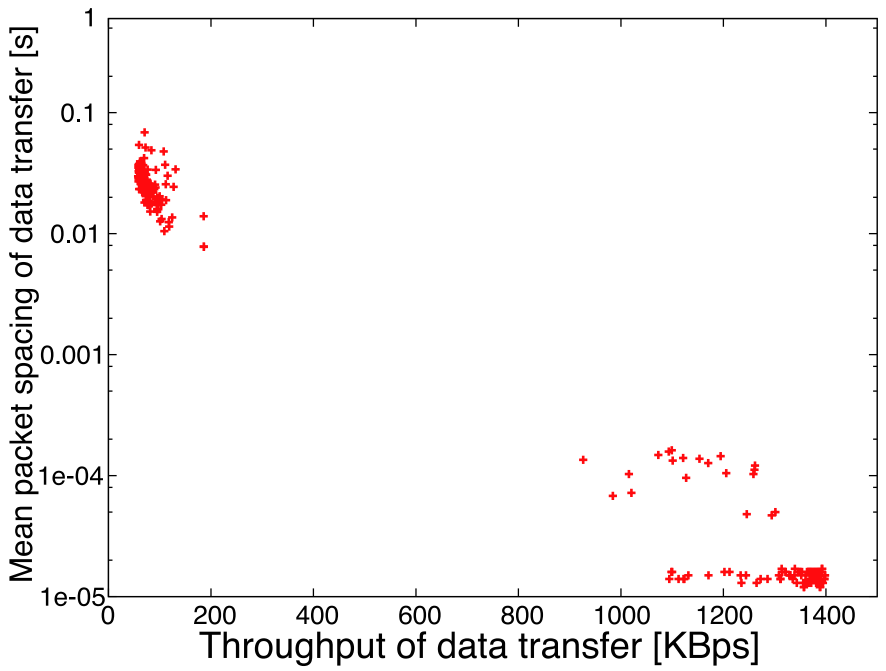


FIGURE 4.3: CDF of packet spacing for connection pair at cases A and B.



(a) Probe transfer (32 KB)



(b) Data transfer (16 MB)

FIGURE 4.4: Mean packet spacing and throughput at index *iii*.

4.4.2 Appropriate Probe Size for Prediction Model

After filtering the anomalous cases, we re-evaluate the data set using ρ. The re-evaluation results are shown in Table 4.2. The ρ for the connection pairs from index *i* to *v* was significantly changed, and a 256-KB probe had the highest value of ρ (0.67). The results show that the anomalous cases are affected to the evaluation results. In the virtualized network environment, we should carefully review network measurement. Finally, we thus selected the 256-KB probe for the connection pair to predict the throughput.

TABLE 4.2: Evaluation results without anomalous cases.

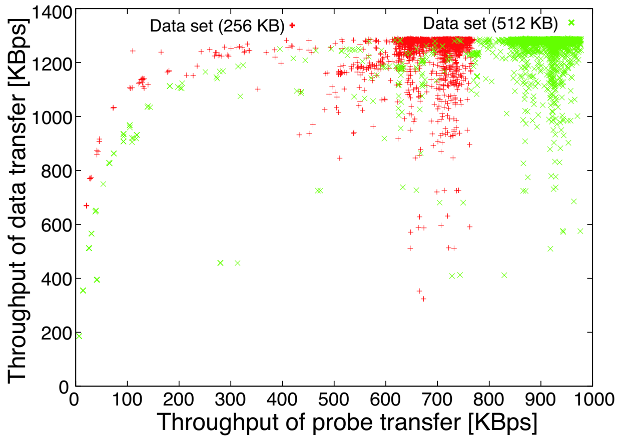
Index	**Probe size [KB]**	**Data size [MB]**	**Rank Cor.** (ρ)
i	16	8	0.31
ii	16	16	0.31
iii	32	16	0.31
iv	64	16	0.32
v	64	32	0.47
vi	128	64	0.42
vii	256	128	0.67
viii	256	256	0.45

4.5 Actual Throughput using Various Probe Sizes

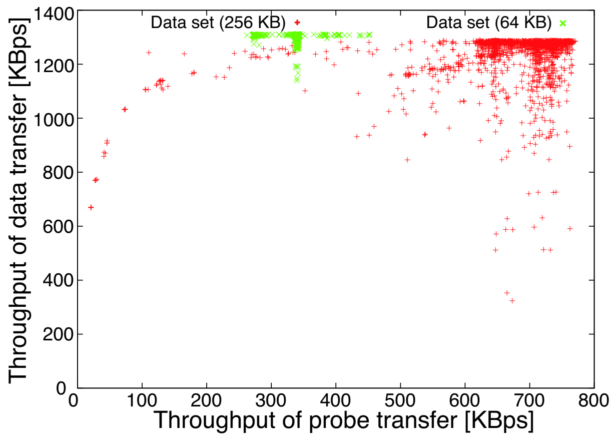
To investigate the effect of probe sizes, two nodes are selected from PlanetLab nodes located in Europe. The mean RTT using ping between the nodes is 0.0477 [s]. We generate connection pairs with the various probe sizes, such as 32 KB, 64 KB, 256 KB, and 512 KB. In the experiments, the data size was unified as 16 MB. To summarize the probe sizes, the 32-KB probe was the best condition with the anomalous case and the 64-KB probe was used for NWS. The 256-KB probe was the best condition without the anomalous cases, and it was selected for our prediction model. Moreover, we select the 512-KB probe to compare the validation of the 256-KB probe. It is twice as large as our probe size.

We gathered approximately 4,000 connection pairs from this node pair over seven days. Actual throughput for these connection pairs is shown in Figure 4.5. With 256-KB probes, the actual throughput monotonically changed, and had noise and non-linear characteristics. Next, no significant changes occurred between the data sets for the 512-KB and 256-KB probes (Figure 4.5(a)). Conversely, actual throughput with 64-KB

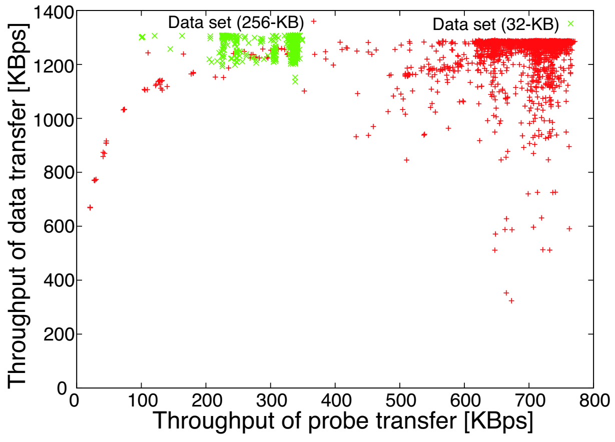
(Figure 4.5(b)) and 32-KB (Figure 4.5(c)) probes is concentrated in certain areas. The small probes, such as 32-KB and 64-KB probes, would be shrunk due to the increased network capacity. As a result, the 256-KB probe has characteristics of a non-linear and continuous monotonic function, and it is appropriate for the prediction model.



(a) 256-KB probe and 512-KB probe



(b) 256-KB probe and 64-KB probe



(c) 256-KB probe and 32-KB probe

FIGURE 4.5: Actual throughput for various connection pairs.

4.6 Conclusion

In this chapter, we show how to adjust experimental setting before the actual prediction. Especially, we have focused on probe size. Noise and non-linear characteristics were founded in our results, and they should be considered for the precise prediction model. Thus, there are no linear characteristics between probe and data transfers. Next, the statistical approach was described to select the appropriate probe size. We showed the reason why Spearman's rank correlation coefficient (ρ) is a valid indicator, and evaluated various sizes for the connection pair through the rank correlation coefficient. Moreover, we presented how the anomalous cases affect the evaluation results. Although there was no high predictability for the connection pair, the small probe sizes had high ρ values. Particularly, the 32-KB probe was the best condition with the anomalous cases. Without the anomalous cases, the 256-KB probe was selected as the probe size. In order to investigate the validation of the selected probe size, we measured the actual throughput with the various probe sizes. While the actual throughput with 256-KB and 512-KB probes monotonically changed and had noise and non-linear characteristics, that with 32-KB and 64-KB probes was concentrated in certain areas. The results presented in this chapter have been published in [24][65].

In the following chapter, our approach for building the precise prediction model is pursued in detail. As a first step, a problem of previous prediction method [17] using CDF of probe and data transfers is introduced. Next, appropriate algorithms are shown to compensate the problem. Then, we build our prediction method with the algorithms. Finally, we compare and discuss prediction results using both prediction methods.

Chapter 5

Analytical Modeling for Network Throughput Prediction

5.1 Overview

In this chapter, a description of the SVR-based predictor and its characteristics are provided. We first introduce and discuss previous prediction methods and how to apply SVMs and SVR into network research in Section 5.2. Second, we describe the composition of data sets, present the actual throughput with the selected probe size on different sites, and discuss their characteristics with the statistics in Section 5.3. Third, we build the previous prediction method [17], and a weak point of previous method is shown with prediction results in Section 5.4. Next, we propose a throughput prediction method, and compare prediction results with those of the previous one in Sections 5.5 and 5.6. Then, we build our prediction method with other probe sizes and non-parametric regression techniques to clarify our assumption for the precise prediction in Sections 5.7 and 5.8. Finally, we conclude with a summary of the main points in Section 5.9.

5.2 Related Work

Here, we introduce and discuss some studies related to the prediction of traffic and throughput to find an alternative prediction method. As a first step, we introduce the studies to predict network traffic. Although their goal is the prediction of network traffic, these methods can help to propose a network throughput prediction method. Again, the aims of our study are to predict network throughput and to improve the accuracy of predicting throughput. Next, we explain and discuss previous throughput prediction methods. Finally, we describe how to apply SVMs and SVR into the network studies.

5.2.1 Prediction of Network Traffic

In order to predict network traffic, there are many prediction methods. Here, we introduce some prediction methods. These methods basically used past network traffic as historical data, and applied mathematical algorithms or machine learning techniques into the historical data to improve the predictability of network traffic. Sang et al. [66] collected continuous traffic data from live networks for a specific period, and subsequently applied Auto-Regressive Moving Average (ARMA) and Markov-Modulated Poisson Process (MMPP) to the traffic data for predicting short to long term traffic fluctuations in network. Then, a traffic prediction method using a Neural Network (NN) is proposed by Cortez et al. [67]. In their method, network topology is used for multivariate strategies, and the method outperformed other forecasting methods (e.g. Holt-Winters). Moreover, Zhao et al. [68] also proposed a traffic prediction method using Artificial Neural Networks (ANNs). Their method performed non-linear mappings between past and present traffic values. However, there is no single ANN form that can capture all the traffic characteristics. In [69], they proposed a hybrid traffic prediction method based on a combination of ANNs and covariation orthogonal prediction. However, their traffic traces focused on wireless local area network only.

5.2.2 Throughput Prediction Methods

Various methods have been proposed for predicting the network throughput. In [70], a scheme that polls Management Information Base (MIB) information of Simple Network Management Protocol (SNMP) from the bottleneck router has been proposed. However, their scheme would be an unsuitable solution because it is not easy to find a bottleneck router. Abusina et al. [71] have proposed Optimistic Network Performance Index (ONPI) and Robust Network Performance Index (RNPI) for the future network throughput. the ONPI corresponds to a range of the best expected network throughput. In contrast, the RNPI corresponds to a range of the lowest expected network throughput. Their heuristic-based method used historical network traffic data, and was similar to the Genetic Algorithm. Next, He et al. [72] proposed Formula-Based (FB) prediction and History-Based (HB) prediction. The formula-based prediction does not rely on historical traffic measurements in an end-to-end path; it would thus be inaccurate in congested paths and dependent on network characteristics. The history-based prediction is based on the Moving Average and Holt-Winter models. Then, Borzemski et al. [73] have developed an application based on data mining algorithms, such as ANNs, decision trees, and transform regression, to predict the throughput in hypertext transfer protocol (HTTP) transactions. They proposed useful parameters for predicting the throughput, but did not clarify why these parameters were useful. Vazhkudai et al. [74] compared several different simple forecasting methods to predict the throughput. Yin et al. [75] proposed a throughput prediction service for many-task computing. It provides users with the optimal number of TCP connections and an estimated time for data transfer. However, their evaluation results focused on Louisiana Optical Network Initiative (LONI) clusters, and they did not investigate the characteristics of experimental networks. Hwang et al. [76] proposed a formula-based predictor with the available bandwidth. Their predictor was accurate in comparison with a predictor based on Amherst model. Their experiments in Local Area Network (LAN) were performed over real network with a simple dumbbell topology while the experiments in Wide Area Network (WAN) were performed through simulation only.

Here, we introduce throughput prediction methods using the probe transfer. Wolski et al. [16] empirically established the basic probe size as 64 KB for the Network Weather Service (NWS). They used the connection pair to predict the throughput of data transfer on NWS and focused on only the connection pair where the probe size was 64 KB and data size was 16 MB. However, they selected the size of the connection pair empirically and generated connection pairs in limited networks, so the probe size might be inappropriate for other networks. Again, we showed that the actual throughput with the 64-KB probe was concentrated on the certain areas in Chapter 4. Moreover, Yousaf et al. [77] reported the requirement of a large-sized probe, but they did not select an appropriate probe size for the Internet. Vazhkudai et al. [78] proposed a linear regression model using a combination of the 64-KB probe and past measurements; their model uses the least squares method. However, there were less data transfers than probe transfers, meaning the data size was not determined precisely. Swany et al. [17] proposed a prediction method using the cumulative distribution function (CDF) of network throughput for probe and data transfers. It computes the CDF of the throughput for probe and data transfers. The throughput for data transfer was predicted by using the CDF of a probe transfer. In particular, we compare in this chapter the prediction results of our prediction method with the prediction method using CDF for the same data sets to show the improved accuracy.

5.2.3 SVM and SVR for Network Research

Support vector machines (SVMs) and support vector regression (SVR) have been used in various network research areas. Bermolen et al. [79] used an SVR for link load prediction. Beverly et al. [80] considered an SVM for predicting round-trip latency. Moreover, Huifang et al. [81] proposed WLAN traffic prediction using an SVM. Mirza et al. [18] have proposed a throughput prediction method using an SVR that combines prior data transfers and measurements of network metrics, such as packet loss, queuing delay, and available bandwidth. However, our method depends only on measurements of the connection pair. Thus, our method uses bivariate data while their method uses multivariate data. In their method, a radial basis function (RBF) for the kernel trick is used to consider non-linear and multivariate regression. They used their laboratory testbed [18] for passive and active measurement of these network metrics. In evaluations, the prediction results with the passive measurement were more accurate than those with the active measurement because of the accurate network metrics for the passive measurements. However, the network metrics are normally undisclosed to users, and it is hard to estimate them on end nodes precisely. Next, they used the Resilient Overlay Networks (RON) testbed [82] to evaluate their method with active measurement on the Internet, but nodes that should have little or no other CPU or network load were restricted and the operating system was also limited to FreeBSD 4.7 for the active measurement of the network metrics. Thus, these limitations may be unsuitable for evaluations on the Internet.

5.3 Data Sets for Throughput Prediction

In this section, we investigate the traffic characteristics through Chapters 2 and 4 with different nodes on the virtualized network testbed for gathering training data sets. As a first step, we introduce the characteristics of selected node pairs on the virtualized testbed. Next, measurement results using the connection pair are shown and their features are also described. Finally, we present input data sets that consist of two types to evaluate prediction results on the same conditions.

5.3.1 Training Data Sets

For training data sets, we empirically selected six pairs of nodes from PlanetLab nodes located in both North America and Europe, which we refer to the pairs as (α, β), (γ, δ), (ϵ, ζ), (η, θ), (κ, λ), and (μ, ν). The geographic location and mean RTT using ping for all the pairs are shown in Table 5.1. We simultaneously generated connection pairs at the sender every 5 minutes. This time we used the 256-KB probe and the 16-MB data. Again, the 256-KB probe had the highest predictability. If the measured size is smaller than expected or if the transfer time is more than 5 minutes, we judge at the receiver that the measurement has failed. Thus, the network throughput was measured using the connection pair. We gathered training data sets for all the node pairs over seven days. There were no the anomalous cases in the training sets. The statistics for the training sets are shown in Table 5.2. The actual throughput for the connection pair at (η, θ) is described in Figure 5.1. It is shown to be non-linear with noise. Thus, probe and data throughputs are monotonically changed. Again, we described these characteristics in Chapter 4. Finally, we should consider noise and non-linear characteristics for the precise prediction model.

TABLE 5.1: Geographic location and mean RTT at node pairs.

Node name	**Geographic location**	**Node pair** (arrow is transfer direction)	**Mean RTT [s]**
α	Europe	$\alpha \leftarrow \beta$	0.0283
β			
γ	Europe	$\gamma \leftarrow \delta$	0.0477
δ			
ϵ	Europe	$\epsilon \leftarrow \zeta$	0.0511
ζ			
η	North America	$\eta \leftarrow \theta$	0.0370
θ			
κ	North America	$\kappa \leftarrow \lambda$	0.0598
λ			
μ	North America	$\mu \leftarrow \nu$	0.0392
ν			

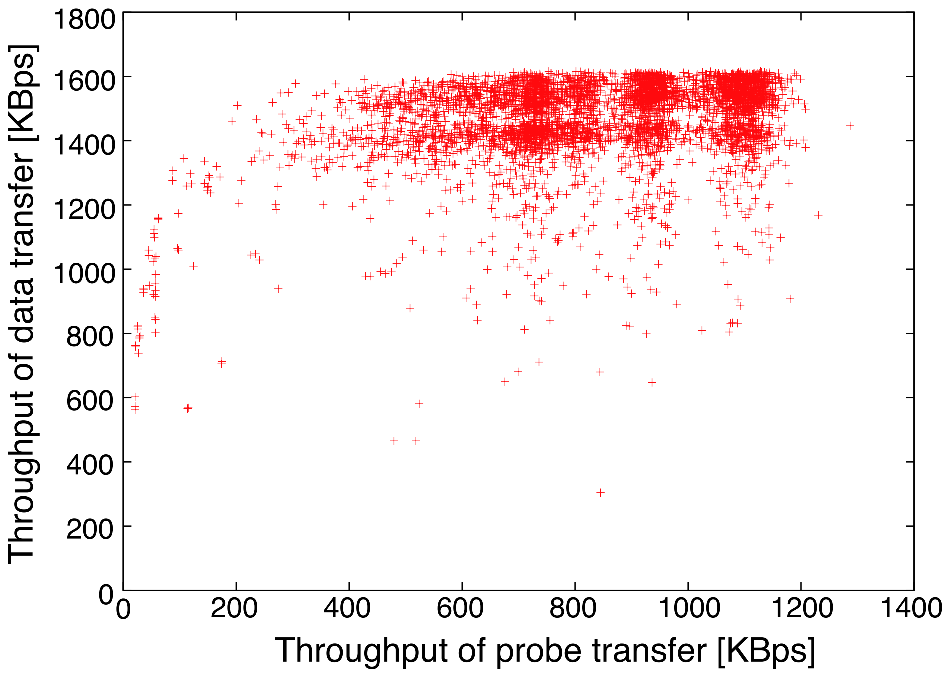


FIGURE 5.1: Actual throughput for connection pair at node pair (η, θ).

TABLE 5.2: Statistics of training data sets. (NT is network throughput.)

Node pair	**Min NT [KBps]**	**Mean NT [KBps]**	**Max NT [KBps]**	**Total counts**
α, β	138.1	2008.0	2107.8	4705
γ, δ	323.7	1280.5	1290.7	3690
ϵ, ζ	0.2	1140.9	1168.0	4535
η, θ	304.3	1467.6	1616.3	4941
κ, λ	56.3	975.0	1032.1	4936
μ, ν	845.9	1335.4	1512.3	5340

5.3.2 Input Data Sets

To evaluate prediction methods, the input data sets per node pair were collected over 36 hours. The actual throughput of the input data set at node pair (η, θ) is shown in Figure 5.2. The actual throughput widely fluctuated with noise. Prediction results under an unstable network state are more important than those under the stable one. If the network state is stable or stationary, we do not have to predict the network throughput. However, the scale and the bandwidth of networks are rapidly increasing and the network state is dynamically changing. To evaluate the prediction results under an unstable network state, we determined the mean throughput of a probe transfer as a threshold value, and additional input data sets consisted of the actual throughput below the threshold value. The composition of the data sets is described in Table 5.3. For example, at node pair (α, β), the number of connection pairs at the input set is 1134 and that at the additional set is 328.

TABLE 5.3: Composition of input data sets. (NT is network throughput.)

Node pair	Mean NT [KBps]	Total counts	Below mean
α, β	1116.0	1134	328
γ, δ	699.2	779	146
ϵ, ζ	586.0	1130	325
η, θ	760.5	929	378
κ, λ	364.9	654	259
μ, ν	656.5	992	377

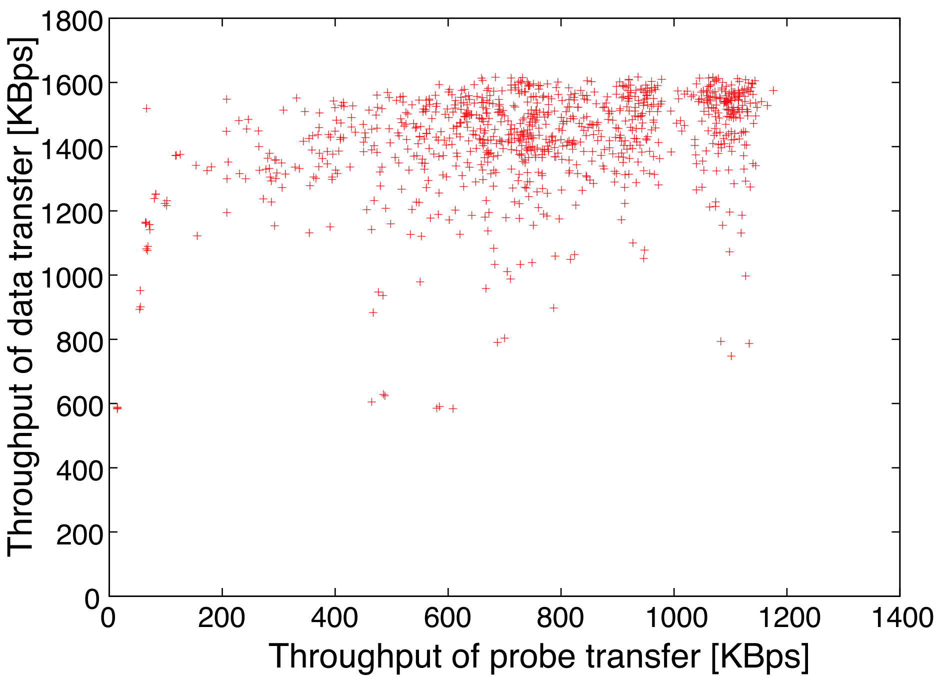
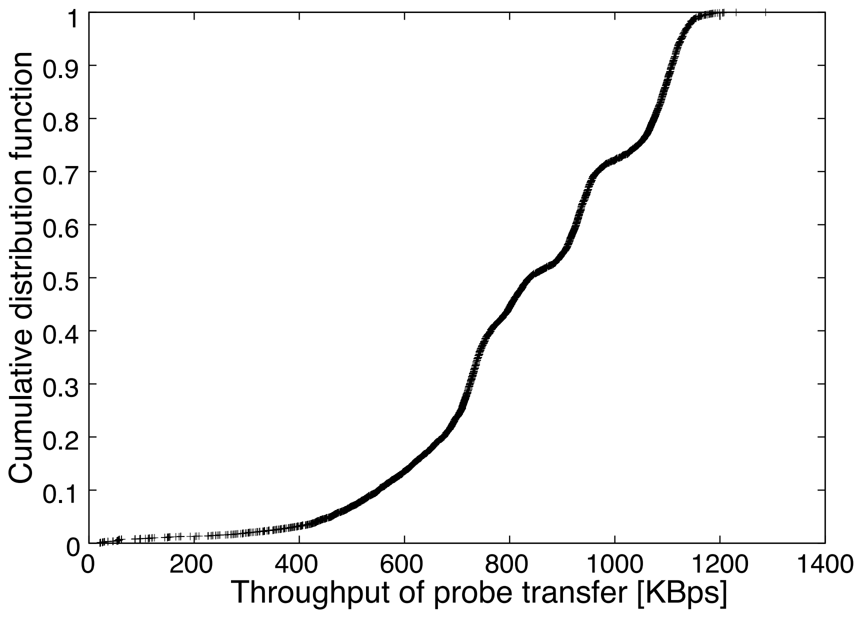


FIGURE 5.2: Actual throughput of input data set at node pair (η, θ).

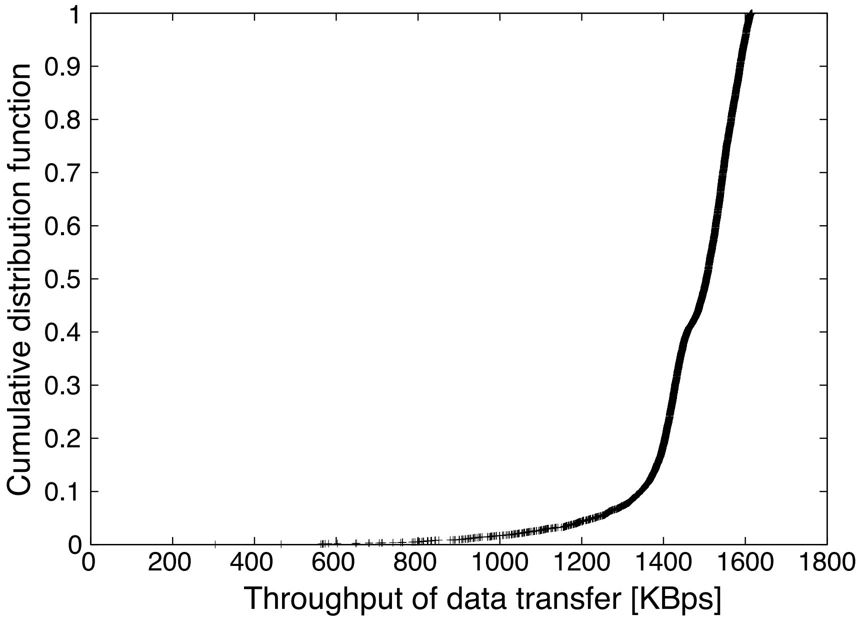
5.4 Previous Prediction Method (CDF-based Predictor)

5.4.1 Building CDF-based Predictor

While other predictors require network metrics, such as RTT, packet loss, and so on, it is possible to build the CDF-based predictor [17] using only the connection pair. Thus, it is appropriate for the comparison of prediction results under the same condition. We first built the CDF-based predictor with the training sets to evaluate whether this predictor produces precise prediction results. It computed the CDF of throughput for probe and data transfers. The throughput for data transfer was predicted by using the CDF of a probe transfer. If there is no noise, the throughput can be predicted precisely. The CDF of the connection pair at node pair (η, θ) is shown in Figure 5.3. In the previous work [17], the curve shape of the CDF of the connection pair was similar, but that of the CDF at all node pairs was different. The cause of the different shape is noise.



(a) CDF of probe transfer



(b) CDF of data transfer

FIGURE 5.3: CDF of connection pair at node pair (η, θ).

5.4.2 Prediction Results

Prediction results of the CDF-based predictor at node pair (κ, λ) are shown in Figure 5.4. Because the CDF-based predictor deals with all data, the results are far from those for the input data set. Thus, a major cause of the difference is noise. Moreover, the difference between the actual throughput and the predicted throughput becomes large when the probe throughput is below the mean probe throughput (364.9 KBps). Therefore, the prediction results would be inaccurate under an unstable network state. The other results are similar to the above results. We should consider the noise and monotonicity between probe and data throughputs for a precise prediction model.

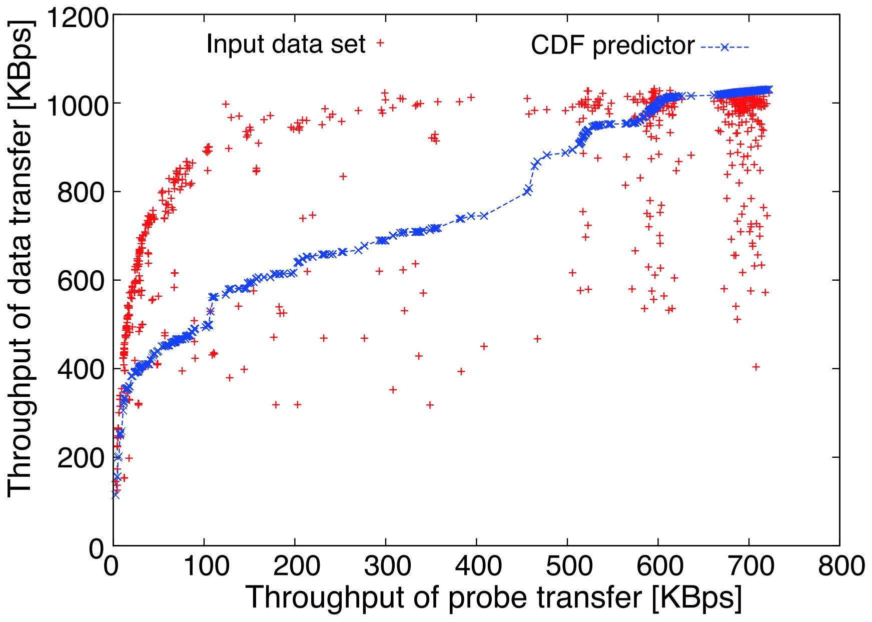


FIGURE 5.4: Prediction results of CDF-based predictor at node pair (κ, λ).

5.5 Proposed Prediction Method (SVR-based Predictor)

5.5.1 SVR Overview

Support vector regression (SVR) is a version of a support vector machine (SVM) [83] for regression. The concept of SVR is to maximize margins. Assume we have a training data set $\{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_i, y_i), ..., (\boldsymbol{x}_l, y_l)\} \in \boldsymbol{R}^n \times \boldsymbol{R}$, where $\boldsymbol{R}^n$ is the space of the input features $\boldsymbol{x}_i$, and y_i is a symbol value. Here, we give an overview of two types of SVR: ϵ-SVR [83] and ν-SVR [84]. ϵ-SVR finds a function $f(\boldsymbol{X})$ that approximates future values accurately. The function is defined as

$$f(\boldsymbol{X}) \;=\; \boldsymbol{w}\boldsymbol{\phi}(\boldsymbol{X}) + b \tag{5.1}$$

where $\boldsymbol{w} \in \boldsymbol{R}^n$, $b \in \boldsymbol{R}$, and $\boldsymbol{\phi}$ is a non-linear transformation from $\boldsymbol{R}^n$ to high-dimensional spaces. An ϵ-insensitive loss function is used to measure an empirical error and is defined as

$$L_\epsilon(f(\boldsymbol{x}_i), y_i) = \begin{cases} 0, & \text{if} |f(\boldsymbol{x}_i) - y_i| \le \epsilon \\ |f(\boldsymbol{x}_i) - y_i| - \epsilon, & otherwise \end{cases} \tag{5.2}$$

ϵ-SVR can be written as

$$min \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \tag{5.3}$$

where C is a weight parameter. The constant $C > 0$ is used to determine the trade-off between training error and model flatness. Slack variables ξ_i and ξ_i^* are allowed to lie outside of an ϵ-insensitive tube. Thus, ϵ-SVR calculates the distance of data points on the tube to determine the shape of the tube. However, the shape of the tube would be changed inappropriately if there was an outlier, such as noise. ν-SVR is a modified version of ϵ-SVR. It has the advantage that a parameter ν, which replaces C, can be interpreted as both an upper bound on the fraction of margin errors and a lower bound on the fraction of support vectors. ν-SVR can be written as

$$min \frac{1}{2}\|\boldsymbol{w}\|^2 - C\left(\nu\epsilon + \frac{1}{n}\sum_{i=1}^{n}(\xi_i + \xi_i^*)\right) \tag{5.4}$$

In ν-SVR, there is no calculation of distance on the outside tube. Even if there is an outlier, we can determine the appropriate shape of the tube. We thus selected ν-SVR for our prediction method. We can apply the kernel trick [85] in SVR without ever having to compute the mapping explicitly. The value of the kernel is equal to the inner product of two vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in the feature space $\boldsymbol{\phi}(\boldsymbol{x}_i)$ and $\boldsymbol{\phi}(\boldsymbol{x}_j)$, that is, $\boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\phi}(\boldsymbol{x}_i) \cdot \boldsymbol{\phi}(\boldsymbol{x}_j)$. Commonly used kernel functions are linear, polynomial, and radial basis. In our prediction method, we use the polynomial function to consider the non-linear characteristics of traffic. It is given by

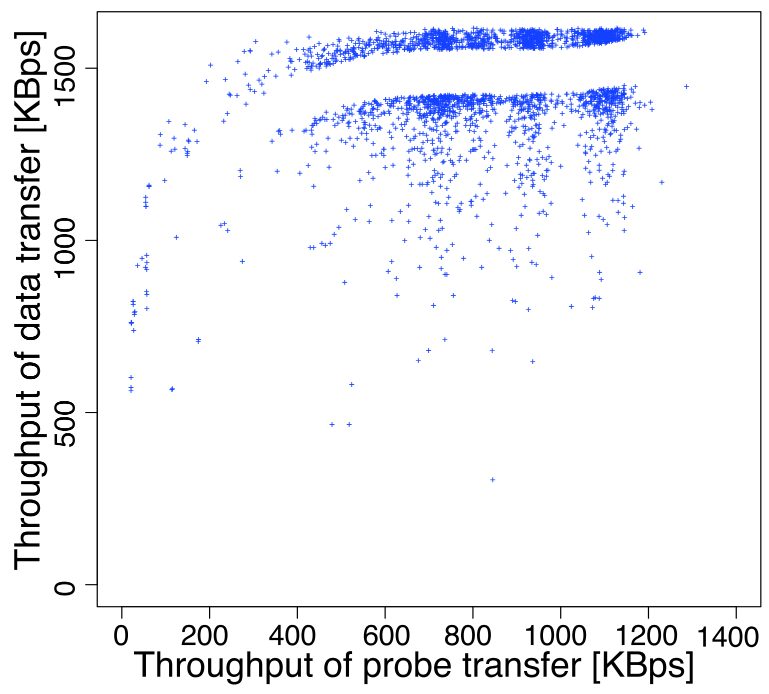
$$\boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) \;=\; (< \boldsymbol{x}_i \cdot \boldsymbol{x}_j > +1)^d \tag{5.5}$$

where d is degrees.

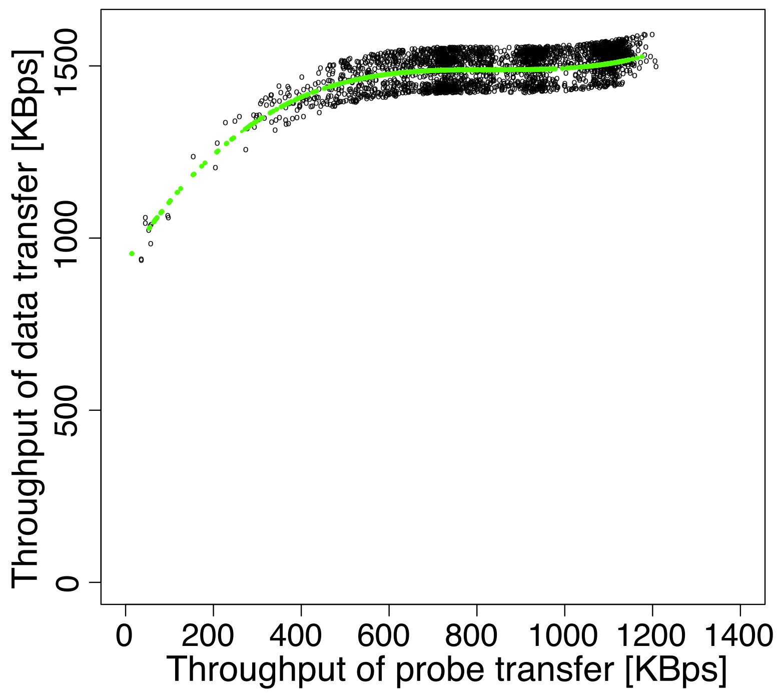
5.5.2 Building SVR-based Predictor

We introduce here our prediction method using ν-SVR and the polynomial kernel. A linear regression curve for throughput prediction would be inappropriate. Various types of traffic, such as mice and elephants [62][64], co-exist in current networks, and spikes that correspond to large and abrupt throughput are occasionally caused by the elephants. The distribution of traffic fluctuation is close to long-tail by the above characteristics. In particular, the marginal distribution of the traffic is not Gaussian [62]. Again, the actual throughput was changed monotonically and there was noise. Then, ϵ-SVR would be inappropriate because it calculates the distance of data points for the shape of the tube. The proposed method uses ν-SVR to deal with noise. The radial basis function (RBF) has been introduced for the purpose of function interpolation, and it can also be used for non-linear characteristics. In comparison with the polynomial kernel, the RBF would be undesirable when there is noise or a paucity of data. Because of this, we apply the polynomial kernel of degree 3 into the proposed method to consider a non-linear and continuous monotonic function. Later, we will discuss whether our selection for the kernel degree is suitable or not.

The SVR-based predictor for node pair (η, θ) is shown in Figure 5.5. The number of support vectors is 2472, and the other data points are used for the tube of the regression curve. Although there is noise, the SVR-based predictor reflects the characteristics of a non-linear and continuous monotonic function. The other node pairs are similar to the above case. The e1071 package [86] in R [87] is used for the predictor. It offers an interface to the libsvm library [88], which is a popular SVM tool. The other parameters in the package are set to the default values.



(a) Support vectors for ν-SVR



(b) Tube of regression curve

FIGURE 5.5: SVR-based predictor at node pair (η, θ).

5.6 Comparison of Prediction Results

To evaluate the accuracy of an individual throughput prediction result at the predictors, we used the relative prediction error (RPE) [72], which is defined as

$$RPE = \frac{\hat{R} - R}{min(\hat{R}, R)} \tag{5.6}$$

where $\hat{R}$ is the predicted throughput and R is the actual throughput. We show the fraction of RPE within 10% or less, which is written as

$$Fraction\, of\, RPE = \frac{\sharp\{\boldsymbol{r} | -0.1 < RPE(\boldsymbol{r}) < 0.1\}}{\sharp\{\boldsymbol{r}\}} \tag{5.7}$$

where $\boldsymbol{r}$ is the input set. If the RPE is high, prediction error is high. However, the fraction of RPE is different from the RPE. For example, the fraction of RPE within 10% or less is the percentage of RPE within 10% or less. Thus, it means that a predictor is accurate if the fraction of RPE is high.

To evaluate the entire input data sets and the entire additional input data sets, we used the root-mean-square error (RMSE). It provides an error for the entire input sets. The minimum error value would be one key criterion in selecting a precise predictor.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{R} - R)^2} \tag{5.8}$$

where n is the number of connection pairs at the input set, and $\hat{R}$ and R are the same as for RPE. To summarize, we compare the individual prediction result through RPE and the entire prediction results through RMSE. The fraction of RPE within 10% or less for the input data sets is shown in Figure 5.6. At node pair (μ, ν), 49.8% of the CDF-based predictor has an RPE of 10% or less while 89.3% of the SVR-based predictor has an RPE of 10% or less. In the other input data sets, the fraction of RPE with SVR was higher than that with CDF. Moreover, the RMSE with SVR (Table 5.4) was also smaller than that with CDF. From these results, the SVR-based predictor is more precise than the CDF-based predictor. Next, prediction results at node pair (η, θ) are shown in Figure 5.7. The regression curve of the SVR-based predictor is more accurate than that of the CDF-based predictor. Thus, the noise had little effect on the SVR-based predictor. The fraction of RPE within 10% or less and the RMSE for the additional sets are shown in Figure 5.8 and Table 5.5 respectively. For the additional sets, the fraction of RPE with SVR was higher than that with CDF. Although the fraction of RPE with the SVR-based predictor decreased, the range of the drop was small in comparison with that of the CDF-based predictor. In the fraction of RPE with CDF, the range of the drop at node pair (κ, λ) was 33.9%, the largest value. Furthermore, the RMSE value with SVR was also smaller than that with CDF. These results are sufficient to show that our SVR-based predictor is precise, robust, and better performing than the CDF-based predictor.

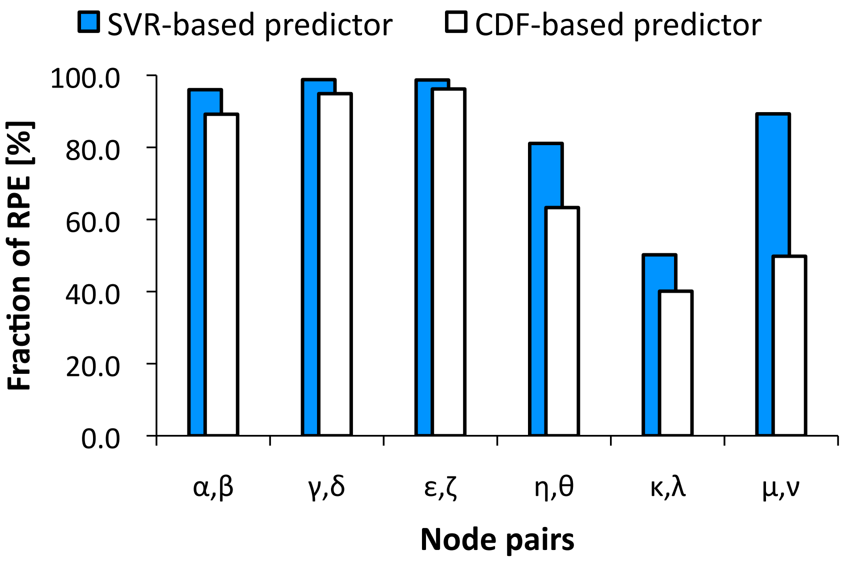


FIGURE 5.6: Fraction of RPE within 10% or less for input data sets.

TABLE 5.4: Root-Mean-Square error of input data sets.

Node pair	SVR-based predictor	CDF-based predictor
α, β	94.6	113.3
γ, δ	27.0	51.4
ϵ, ζ	40.9	79.7
η, θ	155.7	187.5
κ, λ	168.1	203.0
μ, ν	102.7	157.2

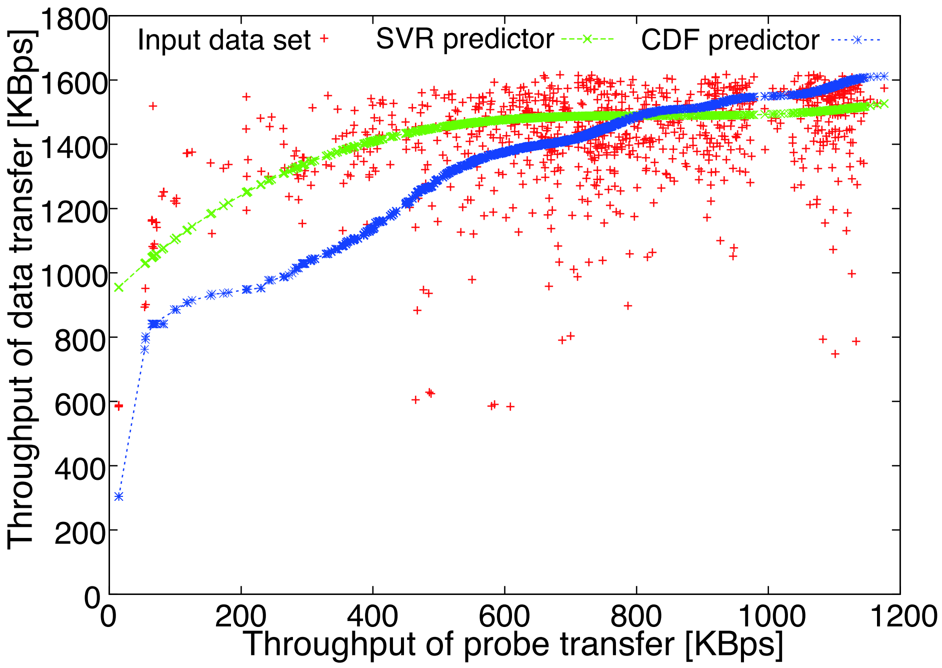


FIGURE 5.7: Prediction results at node pair (η, θ).

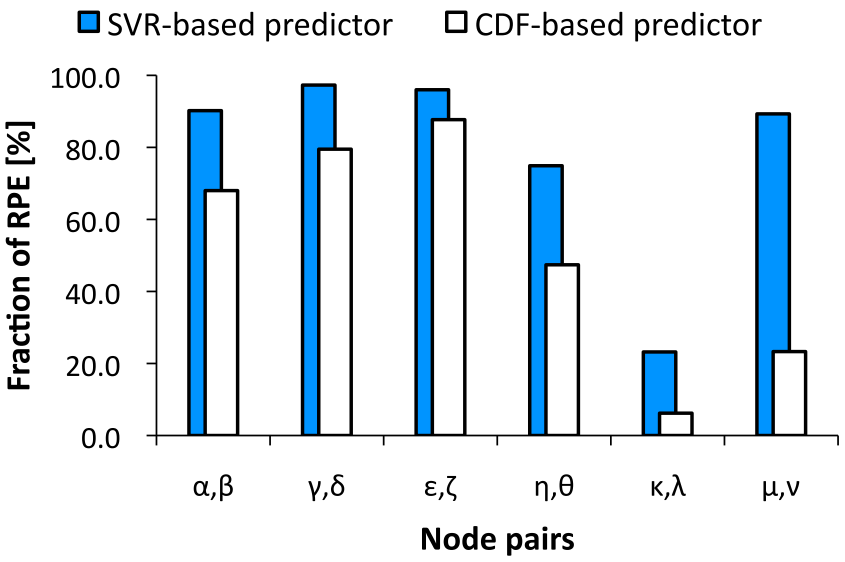


FIGURE 5.8: Fraction of RPE within 10% or less for additional input data sets.

TABLE 5.5: Root-Mean-Square error of additional input data sets.

Node pair	SVR-based predictor	CDF-based predictor
α, β	137.7	180.3
γ, δ	35.0	98.2
ϵ, ζ	56.1	137.7
η, θ	183.0	230.0
κ, λ	182.6	251.9
μ, ν	100.8	192.5

While the CDF-based predictor deals with all data that include noise, our SVR-based predictor uses only characteristic features in the training set. This explains its better prediction results. The regression curve of the SVR-based predictor at node pair (κ, λ) (Figure 5.9) was also closer to the input data set in comparison with that of the CDF-based predictor. Because computational resources and the I/O device of the node on the virtualized testbed are shared by many slices, the sharing can affect the prediction results. Moreover, a congested network state for the node pair can also affect the prediction results. We should clarify what effects lead to the changes in the prediction results, and investigation of this is one of our future works.

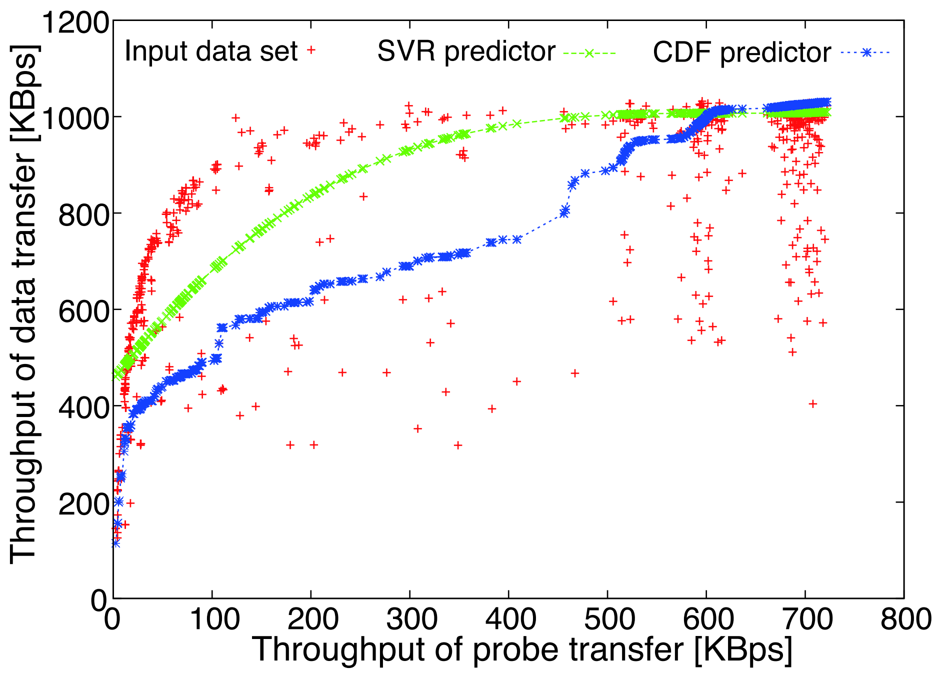


FIGURE 5.9: Prediction results at node pair (κ, λ).

To investigate the adequacy of degree 3, we performed the same experiments by varying the degree from 2 to 5. We omitted the polynomial kernels of degree 6 and above because the regression curve could be fitted to a complicated curve, consequently resulting in overfitting. Moreover, it is time-consuming to determine the regression curve with kernels of high degree. The evaluation results are summarized in Table 5.6. These results show that there are no significant differences in the RPE, except in the case at node pair (κ, λ). Thus, in this study, we concluded that the 3-degree polynomial kernel, which is the default value, is a reasonable choice for our prediction method. Next, from the results obtained with degree 5 at node pair (κ, λ), we found that the 5-degree kernel could achieve better RPE results than the 3-degree kernel. Figure 5.10 shows the regression curves of both 3- and 5-degree kernels. From this figure, we observe that although both the curves do not fit ideally, the discrepancy is small when we use the 5-degree kernel. In future, we intend to investigate the reason why kernels of higher degree can outperform in such cases.

TABLE 5.6: Fraction of RPE of degrees for input data sets. (Deg. is degree).

Node pair	**Fraction of RPE [%]**			
	Deg. 2	**Deg. 3**	**Deg. 4**	**Deg. 5**
α, β	96.0	96.0	96.0	96.0
γ, δ	98.8	98.8	98.8	98.5
ϵ, ζ	98.7	98.7	98.3	98.3
η, θ	81.5	81.1	80.2	79.9
κ, λ	48.9	50.2	51.8	53.8
μ, ν	89.0	89.3	89.6	89.6

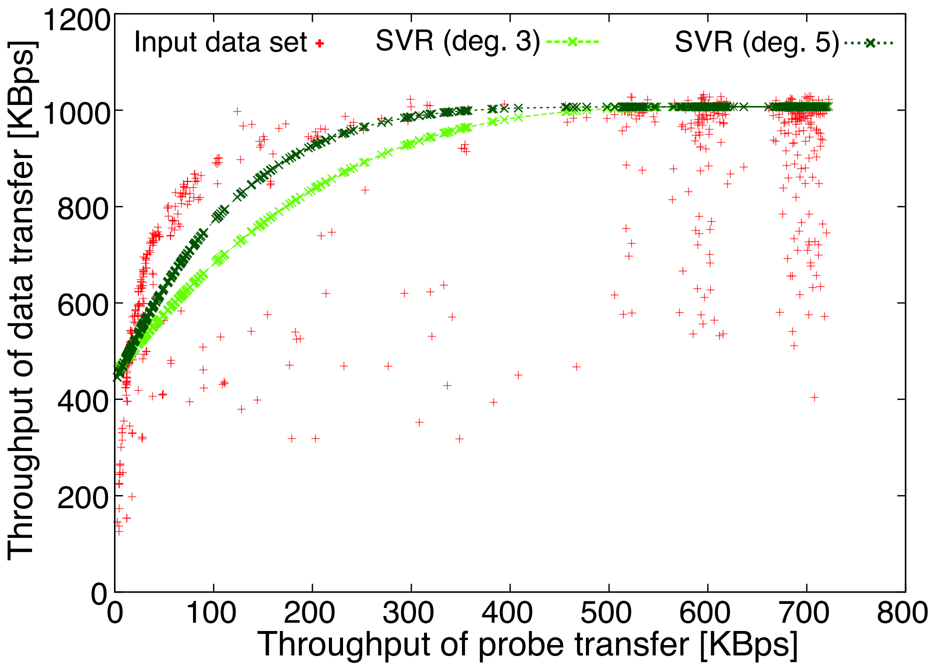


FIGURE 5.10: Prediction results with degree 3 and 5 at node pair (κ, λ).

To summarize, the actual throughput for the connection pair had noise and non-linear characteristics, and the CDF-based predictor was unsuitable for considering the above characteristics. In our prediction method, ν-SVR and the polynomial kernel are used to deal with a non-linear and continuous monotonic function. These lead to the improved prediction results. In the evaluation, we showed that our proposed method is better performing than the CDF-based predictor through RPE and RMSE.

5.7 Issue of Probe Size

Our assumption is that the prediction model is precise with the probe size that has high predictability. Thus, the assumption should be clarified. We here compare prediction results of other probes with those of the proposed probe. To investigate the effect of the probe size on the SVR-based predictor, we increased the probe size to 512 KB, and decreased it to 64 KB for the connection pair. We showed features of these probe sizes in Chapter 4.

5.7.1 SVR-based Predictor with 512-KB Probe

To investigate the effect of a larger probe size, we increased the probe size to 512 KB. It is twice as large as the selected probe size. The numbers of training data sets and input data sets were similar to those for the 256-KB probes. Because the probe size was increased from 256 KB to 512 KB, the actual throughput for probe transfer increased. However, no significant changes occurred between the training sets for the different-sized probes. Again, we showed the characteristics in Chapter 4. The actual throughput of the training sets at node pair (η, θ) is depicted in Figure 5.11. The actual throughput is similar to the training set with 256-KB probes. Thus, the connection pair showed characteristics of a non-linear and continuous monotonic function.

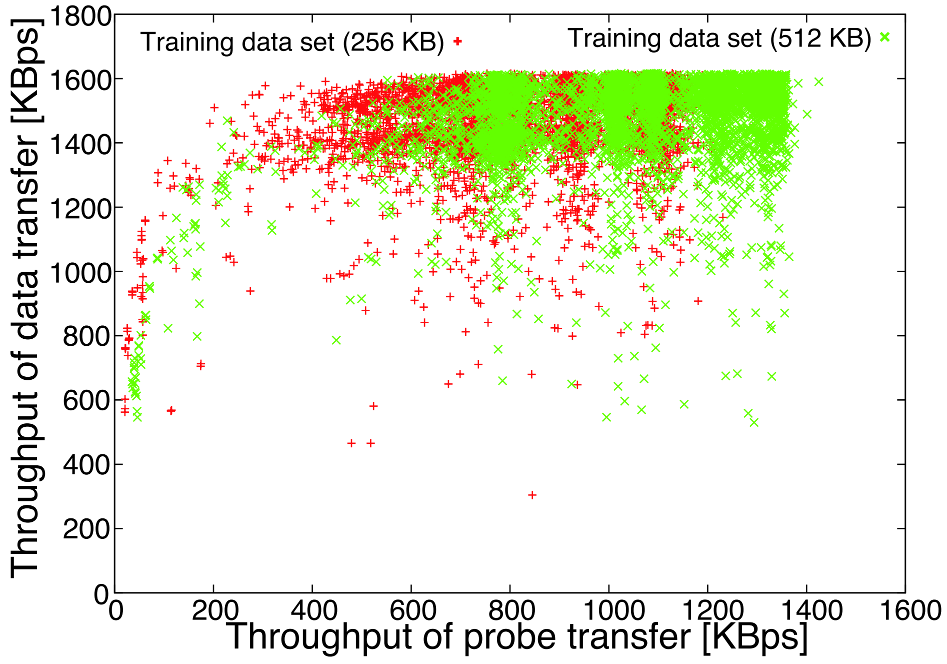
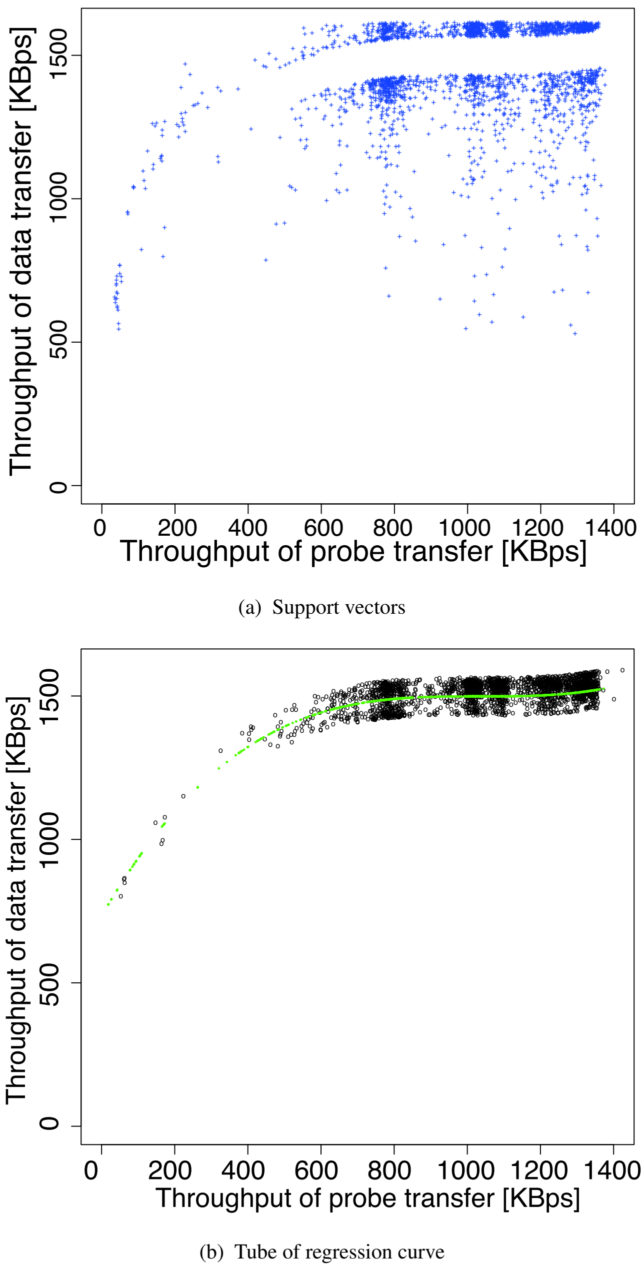


FIGURE 5.11: Actual throughput of training data sets at node pair (η, θ).

We built the SVR-based predictor with the same parameters as previously described. The predictor at node pair (η, θ) is shown in Figure 5.12. It is remarkably similar to the predictor with 256-KB probes. The other node pairs also show the same form. Next, we evaluated the node pairs with RPE to clarify the individual throughput prediction result. The fraction of RPE within 10% or less with 512-KB probes is shown in Figure 5.13. There are no significant changes although the accuracy at node pair (κ, λ) is slightly higher than that with 256-KB probes. Moreover, the RMSE with the 512-KB probe (Table 5.7) was also similar to that with the 256-KB probe. From these results, there is no advantage to increasing the probe size for the prediction accuracy.



(a) Support vectors

(b) Tube of regression curve

FIGURE 5.12: SVR-based predictor with 512-KB probe at node pair (η, θ).

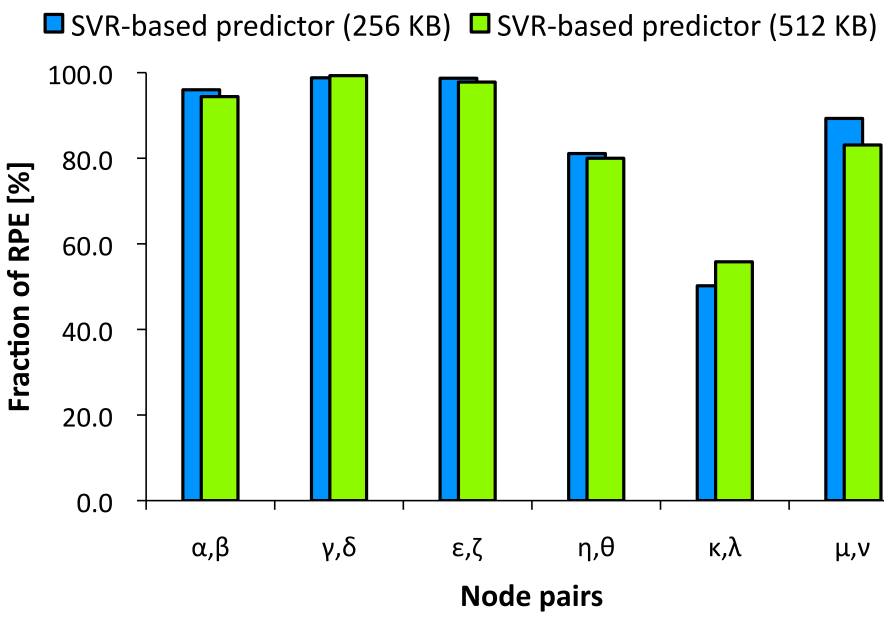


FIGURE 5.13: Fraction of RPE within 10% or less with 512-KB probes.

TABLE 5.7: Root-Mean-Square error with 256-KB and 512-KB probes.

Node pair	**256-KB probe**	**512-KB probe**
α, β	94.6	85.2
γ, δ	27.0	20.0
ϵ, ζ	40.9	43.6
η, θ	155.7	153.2
κ, λ	168.1	190.0
μ, ν	102.7	109.6

5.7.2 SVR-based Predictor with 64-KB Probe

To investigate the effect of a small probe size, we changed the probe size to 64 KB. Again, it was proposed by Wolski et al. [16] and used for NWS. We gathered training sets without node pair (ϵ, ζ) because the site of node ζ dropped out PlanetLab Consortium during the experimental period. The number of training sets was similar to that with 256-KB probes. Although we only decreased the probe size from 256 KB to 64 KB, the actual throughput with 64-KB probes was different from that with 256-KB probes. The actual throughput at node pair (γ, δ) is shown in Figure 5.14. In this case, the number of connection pairs in the training set is 5083. The actual throughput with 256-KB probes is fluctuated, and we can find an appropriate regression curve for the prediction. However, the actual throughput with 64-KB probes is concentrated in certain areas, and the probes cannot reflect the data transfer. This shows that building a precise predictor is difficult. We also eliminated this pair in building the SVR-based predictor. Moreover, we observed considerable noise at the other pairs. The actual throughput at node pair (η, θ) is depicted in Figure 5.15. There was considerable noise at the actual throughput, which becomes an obstacle to finding the appropriate regression curve for the SVR-based predictor.

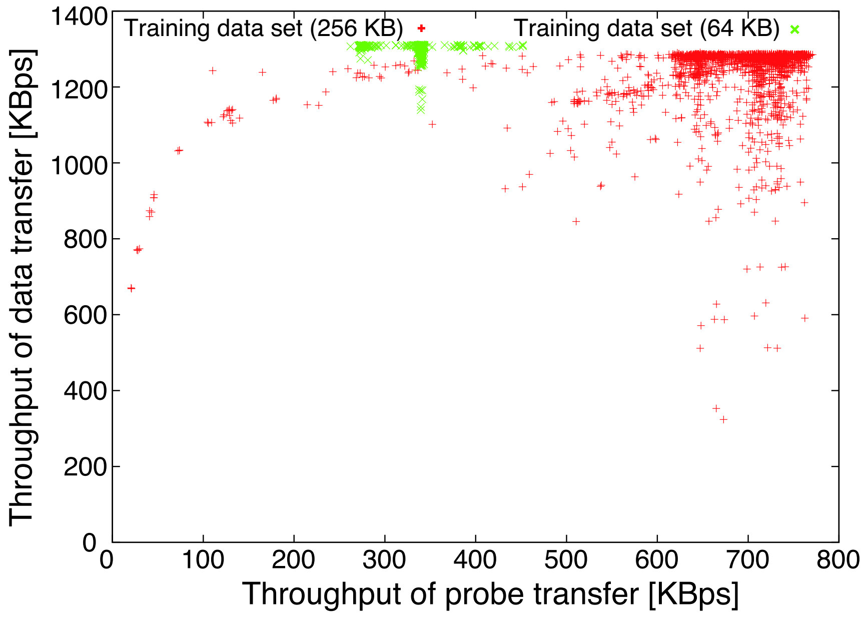


FIGURE 5.14: Actual throughput of training data sets at node pair (γ, δ).

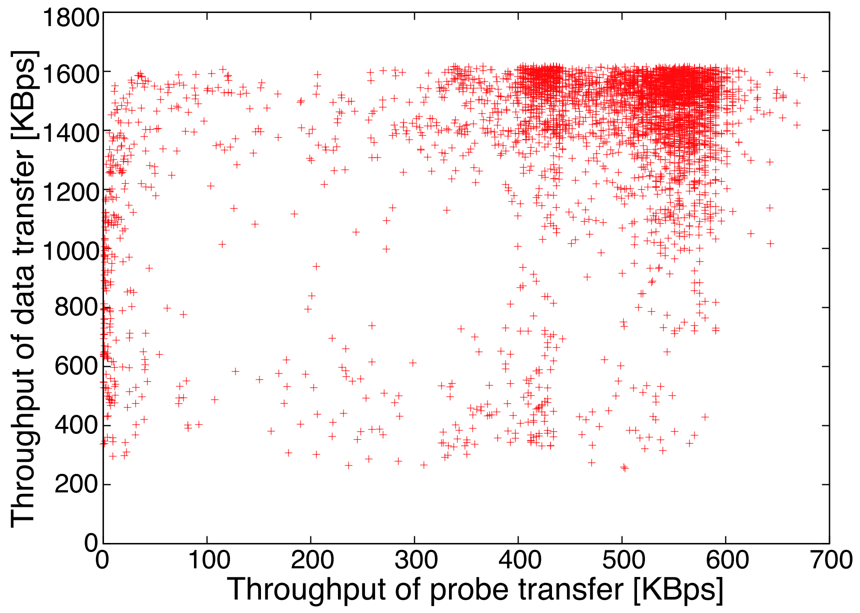


FIGURE 5.15: Actual throughput of training data sets at node pair (η, θ).

We used four node pairs to build the SVR-based predictor, and we evaluated them with RPE. Prediction results at node pair (η, θ) are described in Figure 5.16. The regression curve of the SVR-based predictor cannot represent all of the features in the data set. The fraction of RPE within 10% or less with 64-KB probes is shown in Figure 5.17. Due to the considerable noise, the prediction accuracy at all the node pairs was decreased. The effect of noise is shown in the RMSE with the 64-KB probe (Table 5.8). All of values in the RMSE with the 64-KB probe are dropped in comparison with those with 256-KB probe. To summarize, the actual throughput with 64-KB probes is different from that with 256-KB probes. We observed substantial noise at the actual throughput. This is a major cause of the decreased prediction accuracy.

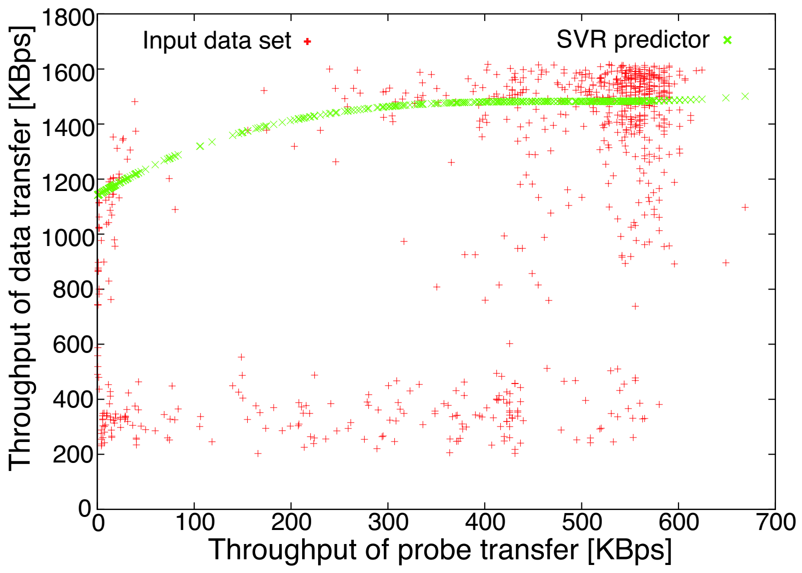


FIGURE 5.16: Prediction results at node pair (η, θ).

TABLE 5.8: Root-Mean-Square error with 256-KB and 64-KB probes.

Node pair	**256-KB probe**	**64-KB probe**
α, β	94.6	192.0
η, θ	155.7	551.9
κ, λ	168.1	297.4
μ, ν	102.7	141.8

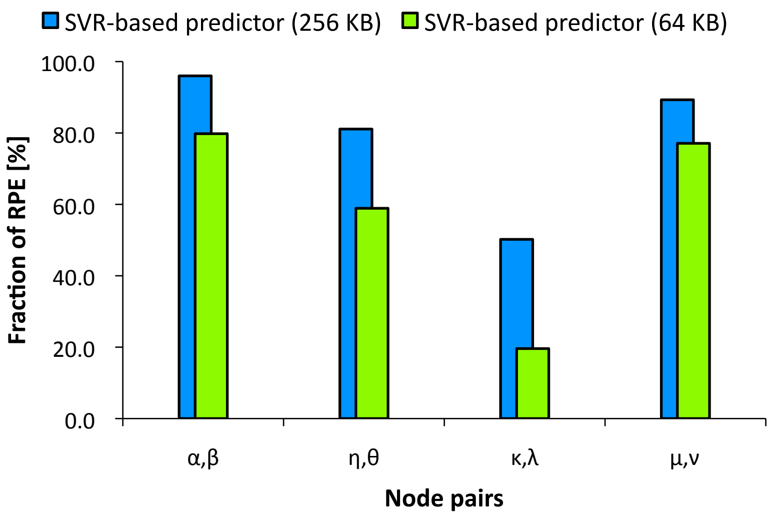


FIGURE 5.17: Fraction of RPE within 10% or less with 64-KB probes.

5.8 Issue of Predictability using Non-parametric Regression Techniques

In the previous sections, we focused on the precise prediction model when the actual throughput has noise and non-linear characteristics, evaluated the prediction results, and presented the validation of our assumption related to the issue of probe size.

In this section, we mainly compare the prediction results using the SVR-based predictor to those using non-parametric regression techniques. As a first step, we present different points of non-parametric regression in comparison with parametric regression. Next, we briefly introduce generalized additive models (GAM) [89][90] and multivariate adaptive regression splines (MARS) [91]. Then, we build prediction models using GAM and MARS for the same data sets that build the SVR-based predictor. Finally, we show and discuss the prediction results using the regression techniques.

5.8.1 Non-parametric Regression Techniques

We here describe what is non-parametric regression and what is different from parametric regression. Then, we introduce and discuss the characteristics of GAM and MARS.

Unlike parametric regression uses a finite set of parameters, non-parametric regression accommodates a very flexible form of regression curve. For example, polynomial regression consists of performing multiple regression with variables $(x, x^2, x^3, ...)$ to find an appropriate regression curve. In other words, parametric regression is based on a finite number of parameters. Contrarily, non-parametric regression provides a versatile method of exploring a general relationship between variables. Moreover, it requires larger samples than those of parametric regression because data must supply model structure as well as model estimation. Two of the most commonly used approaches to non-parametric regression are smoothing splines and kernel regression. Smoothing splines minimize the residual sum of squares, and plus a term which penalizes the roughness of the fit. Kernel regression involves making smooth composites by applying a weighted filter to data. Both methods are useful for many areas of science, economy, and technology.

5.8.1.1 GAM Overview

In order to understand GAM, we first introduce generalized linear models (GLM). It assumes a linear relationship between values of dependent variables and independent variables. Let $\boldsymbol{Y}$ be a dependent variable, and $\boldsymbol{X}_1, ..., \boldsymbol{X}_p$ be p independent variables. The mean of $\boldsymbol{Y}$ is modeled as a linear function of $\boldsymbol{X}_1, ..., \boldsymbol{X}_p$, which is written as

$$\begin{aligned} E(\boldsymbol{Y}) &= f(\boldsymbol{X}_1, ..., \boldsymbol{X}_p) \\ &= \beta_0 + \beta_1 \boldsymbol{X}_1 + ... + \beta_p \boldsymbol{X}_p \\ &= \beta_0 + \sum_{j=1}^{p} \beta_j \boldsymbol{X}_j \end{aligned} \tag{5.9}$$

Given a sample of values for $\boldsymbol{Y}$ and $\boldsymbol{X}$, where $\boldsymbol{X} = (\boldsymbol{X}_1, ..., \boldsymbol{X}_p)$, the estimates of $\beta_0, \beta_1, ..., \beta_p$ are obtained by a particular response function, such as least squares method.

GAM blends properties of GLM with additive models, which is written as

$$\begin{aligned} E(\boldsymbol{Y}) &= f(\boldsymbol{X}_1, ..., \boldsymbol{X}_p) \\ &= s_0 + s_1(\boldsymbol{X}_1) + ... + s_p(\boldsymbol{X}_p) \\ &= s_0 + \sum_{j=1}^{p} s_j(\boldsymbol{X}_j) \end{aligned} \tag{5.10}$$

where s_j are unspecified smooth functions estimated from data. To summarize, GAM replaces the response function with the smooth functions. The gam package [92] in R is used for our prediction model.

5.8.1.2 MARS Overview

MARS is a non-parametric regression technique and can be seen as an extension of linear models that automatically models nonlinearity and interactions between variables. MARS model has the following form:

$$\boldsymbol{Y} = \beta_0 + \sum_{m=1}^{M} \beta_m \boldsymbol{B}_m(\boldsymbol{X}) \tag{5.11}$$

where $\boldsymbol{Y}$ is output variable, $\boldsymbol{X}$ is input variable, β_0 is the coefficient of constant term, M is the number of spline basis functions, $\boldsymbol{B}_m$ is the m^{th} spline basis function, and β_m is the best balance between training error and generalization error. In MARS, the suitable value of M is determined from training data.

MARS builds a regression model in two phases: the forward and the backward pass. Two phases are the same as that used by recursive partitioning trees.

- The forward pass
  In this phase, MARS repeatedly adds the spline basis functions to the model. At each step, it finds the pair of spline basis functions that gives the maximum reduction in the residual sum of squares. This process of adding terms continues until the change in residual error is small to continue or until the maximum number of terms is reached. Again, the maximum number of terms is specified by the user before the model building starts. In the forward pass, the model can be overfitted due to a large number of basis functions.

- The backward pass
  The backward pass prunes the model to build a model with better generalization ability. It removes terms one by one, deleting the least effective term at each step until it finds the best submodel. Model subsets are compared using generalized cross validation (GCV). Finally, the best submodel that has the lowest values of GCV is selected as the regression model.

The earth package [92] in R is used for our prediction model because the term “MARS” is trademarked and licensed to Salford Systems.

5.8.2 Prediction Results

We here present prediction results using GAM and MARS to compare them to the prediction results using the SVR-based predictor. So, we build predictors using GAM and MARS with the training data set of the 256-KB probe. Two types of prediction results using GAM and MARS are found. The former is the similar prediction results using the SVR-based predictor and the latter is the overfitted prediction results. The prediction results at node pair (η, θ) are similar to those using the SVR-based predictor. The results are depicted in Figure 5.18. Similar regression curves are found in comparison with that using the SVR-based predictor. Meanwhile, the prediction results at node pair (κ, λ) are different from the above results. The results are shown in Figure 5.19. The regression curve using GAM is overfitted becasue the model is interpolated by all of data. Thus, the curve is obviously affected by noise. Moreover, the regression curve using MARS (Figure 5.20) is also affected by noise. The predicted throughput is decreased when the throughput of probe transfer is larger than 650 KBps. It is enough to present the overfitted results of MARS.

To summarize, the prediction results using the non-parametric regression techniques did not have the monotonicity between probe and data throughputs. Moreover, the regression curve is more sensitive than the SVR-based predictor because the regression techniques focus on the interpolation between data points.

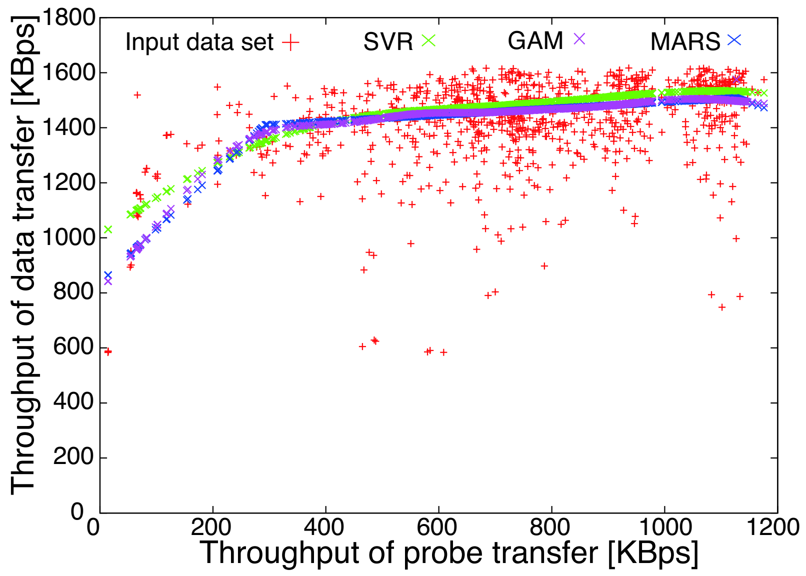


FIGURE 5.18: Prediction results using SVR, GAM, and MARS at node pair (η, θ).

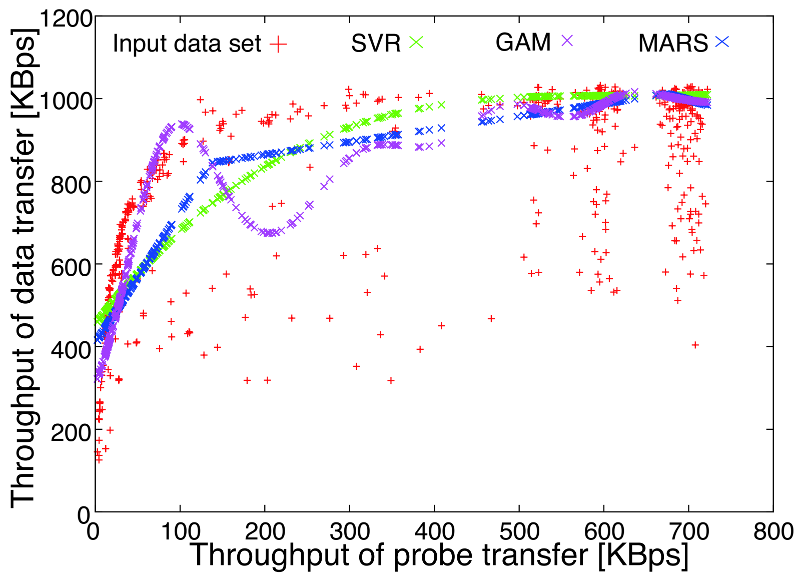


FIGURE 5.19: Prediction results using SVR, GAM, and MARS at node pair (κ, λ).

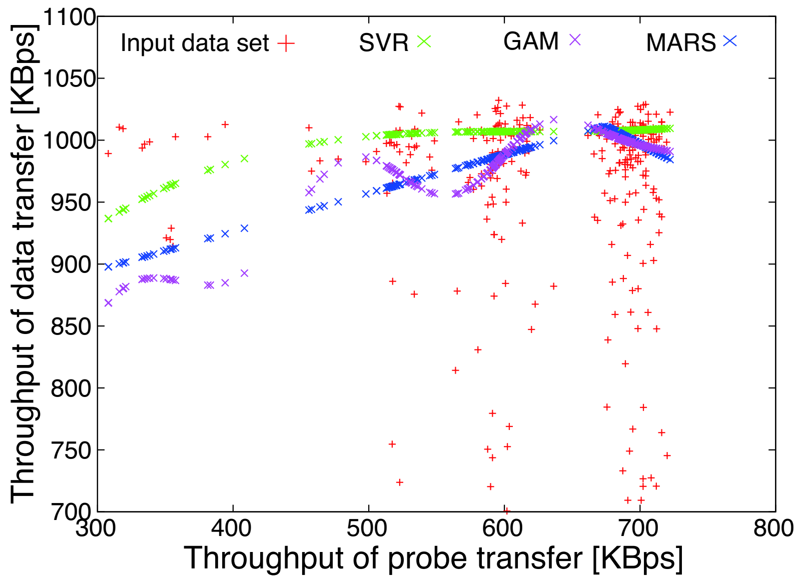


FIGURE 5.20: A part of prediction results using SVR, GAM, and MARS at node pair (κ, λ).

5.9 Conclusion

This chapter shows that the SVR-based predictor can significantly improve the prediction results if the actual throughput has noise and non-liner characteristics. ν-SVR and polynomial kernel are the key elements for the improvements and these elements are represented in the SVR-based predictor. It was established that the SVR-based predictor is more accurate, robust, and suitable than the CDF-based predictor with the same condition. Moreover, issues of suitable probe size are shown and discussed to present the validation of our approach. As a result, the SVR-based predictor based on the connection pair that uses ν-support vector regression and the polynomial kernel to deal with prediction models represented as a non-linear and continuous monotonic function is proposed and illustrated. Our prediction method is enough to answer research questions described in Chapter 1. The SVR-based predictor has been published in [25][26].

In the next chapter, we present how the prediction results contribute to grid scheduling through simulation. First, a simulation method is described with real throughput traces. Next, qualitative evaluation results of the simulation are shown and discussed. Finally, performance implications are illustrated through the evaluation results.

Chapter 6

Performance Implications by Predicting Throughput

6.1 Overview

Through Chapters 2 and 5, we explored a way to predict the network throughput on the Internet. Our prediction method was based on analysis results of Internet traffic, and research issues of Internet traffic characteristics were also discussed for our prediction model. We used machine learning techniques (ν-SVR and polynomial kernel) for precise prediction results. With the same condition, our SVR-based predictor is more accurate, robust, and suitable in comparison with the CDF-based predictor. However, we only focused on the precise prediction and we did not discuss the contribution of prediction results to real applications, such as grid computing. In detail, it is unclear how high accuracy of throughput prediction should contribute to realize efficient scheduling. Moreover, it is also unclear that higher accuracy should always guarantee better scheduling.

In this chapter, we focused on how the predictors affect meta-scheduler performance quantitatively. We first introduce our motivation and some studies related to grid services and schedulers in Sections 6.2 and 6.3. Second, we describe the composition of data sets, build the predictors, and evaluate their prediction results in Sections 6.4 and 6.5. Next, our simulation method is introduced in Section 6.6. Then, the evaluation of simulation results and performance implications are described in Section 6.7 and Section 6.8 respectively. Finally, we conclude with a summary of the main points in Section 6.9.

6.2 Motivation

Modern Scientific experiments, such as high-energy physics, astronomy, and climate modeling, become increasingly data-intensive. The data volumes of such experiments are growing to petascales [93][94][95][96]. In 2007, the Large Hadron Collider (LHC) [94] produces roughly 15 petabytes of data annually. Such data-intensive tasks are processed on geographically distributed resources, and the Internet is often used to share the resources on sites. Grid middleware [97][98] and services [1][2] have been developed to offer efficient available resources to site users.

A meta-scheduler is used to efficiently manage resources on remote sites. It is deployed on a layer above the sites and assigns tasks to the site that has the best resources. The scheduler can reduce the overall processing time of given tasks. Hereafter, we refer to the "scheduler performance" as the amount of time that is saved using the scheduler, compared to another scheduler. The performance of the meta-scheduler can also be enhanced by the efficient use of network resources. The scheduling problems in regard to both computational and communication resources are known as co-allocations [99][100] when dedicated communication links are available among grid sites. However, allocation and reservation of communication resources are difficult when the Internet is used. Under this situation, the throughput prediction on the Internet is promising for improving the scheduler performance. However, it is unclear how high accuracy of throughput prediction should contribute to realize efficient scheduling. Moreover, it is also unclear that higher accuracy should always guarantee the improved scheduler performance. The aim of this chapter is to quantitatively clarify how the predictors can affect the overall processing time of given tasks.

6.3 Grid Services and Schedulers

Many grid services and network-aware schedulers have been developed to improve scheduling performance in grid computing. We here introduce some services and schedulers. The grid harvest service (GHS) [101] was developed by Wu et al., and it provides task scheduling performance prediction. However, network resources are not considered. Grid Service Broker [102] uses network information provided by the NWS [16]. They applied the broker to a data-intensive environment and showed a reduced processing time. However, we have already observed that the 64-KB probe was inappropriate for predicting the network throughput in Chapters 4 and 5. Chinnaiah et al. [7] proposed monitoring architecture with network cost function for grid network. Again, it is not easy to estimate network metrics for the function on the Internet. Moreover, they only used local grid testbeds for their evaluations. Caminero et al. [103] proposed a strategy to perform P2P meta-scheduling in the grid by considering the network characteristics. In their strategy, the nodes forward queries to neighbors using Routing Indices (RI) [104] to find neighbors that are more likely to have required resources. The strategy showed performance improvements through simulations. But, the strategy considered only physical topology, which is not providing most efficient query forwarding, and they did not show how to change the network characteristics on the DataGRID [105] for the simulations. McClatchey et al. [106] proposed a meta-scheduler called Data Intensive and Network Aware (DIANA). It considers network cost for scheduling decision in the grid. The network metrics, such as bandwidth, latency, packet loss, jitter, and anomalies of network links, should be provided for the network cost, i.e., DIANA would be unsuitable for the current Internet. Hsu et al. [107] proposed a selection scheme coupled with P2P co-allocation based on available bandwidth between a client and different servers. However, control overhead is high and the selection scheme requires available bandwidth. Christodoulopoulos et al. [108] have also proposed network resource scheduling. NS-2 [109] was used for their simulation and they showed performance benefits by co-allocating the network and computational resources. However, the above works did not use the real throughput fluctuation traces for their simulations.

For the efficient co-allocations, resource reservation in advance has been considered. In [99][100], network resources were used as distributed resources, and G-lambda [99] and GridARS [100], which provide reservation services for computational and network resources, were proposed. In addition, they extended their service to multiple network domains [110]. However, their testbeds were located on JGN [111]. Thus, networks for the reservation should support Generalized Multiprotocol Label Switching (GMPLS) [112] and provide the network metric to their Network Resource Management (NRM) systems. Kokkinos et al. [113] proposed an information aggregation scheme for resource management in the grid. The aggregation scheme was based on spectral clustering [114], and considered not only computational resources but also network resources, such as delay and bandwidth. In their simulations, however, they only used discretized time availability for the resource reservation in advance. In other words, the network resources were summarized as discretized availability, and it would be hard to fully reflect characteristics of network. Liang et al. [115] proposed Earliest Start Time

Estimator (ESE) for Advance Reservation Service (ARS). Experiments showed the efficiency of ESE. However, it is unclear how to use network resource in ESE. Tomás et al. [6] proposed a network-aware meta-scheduler to consider the case where the reservations are impossible. But, their scheduler only used a mean value of past transfers for estimated transfer time. Experiments were also executed on local grid testbeds.

To summarize these studies, the dedicated infrastructures or disclosure of network metrics are required for their services and schemes. It is too costly and takes a lot of time to construct such infrastructures on the Internet. Moreover, it is hard to fully disclose the network metrics among ISPs or countries. Although an estimation of network metrics is one of alternative solutions, it is too hard to precisely estimate the metrics on end nodes. Moreover, it would be obvious that the imprecise metrics have the negative effects of scheduling. Finally, our approach is different from the studies, thus it does not require hardware for the or the disclosure and it can be applied to the current Internet.

6.4 Measurement Methodology

In this section, we introduce two types of data sets on different nodes. The former is to build the predictors and the latter is to evaluate them and to use our simulation. Here, we clarify actual throughput for the connection pair even if the different nodes are used. The data set is transformed into time-series data to reflect the throughput fluctuation for the simulation. It will be clear how to fluctuate the actual throughput over the time.

6.4.1 Training Data Sets

There are many types of topologies for the grid. We use a star topology for our meta-scheduler simulation. It is a basic hierarchical topology and enough to show the effect of prediction results. In future, we will explore simulations with other topologies, such as P2P based. Our environment was composed as the star topology in which all sites are connected to a central site. A PlanetLab node from North America was empirically selected as the central site. Then, PlanetLab nodes having similar network metrics on different sites from North America were also selected, and are referred to as sites S-0, S-1, S-2, S-3, S-4, S-5, and S-6. Next, connection pairs were generated at the central site every 300 [s] over 7 days when the probe size was 256 KB and the data size was 16 MB. Again, discussions on how to choose proper sizes for both probe and data transfers are presented in Chapter 4. In Figure 6.1, the actual throughput of connection pair is fluctuated and has noise and non-linear characteristics. The other cases are similar to the above case. Again, the actual throughput characteristics were also observed in Chapters 4 and 5. The statistics of learning data sets are shown in Table 6.1.

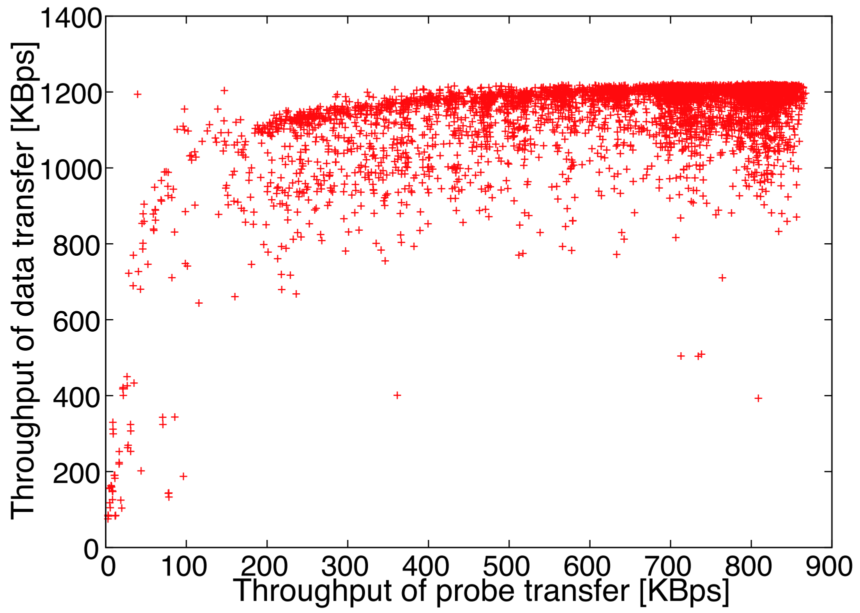


FIGURE 6.1: Actual throughput of connection pair at site S-4.

TABLE 6.1: Statistics of learning data sets.

Site	**Min NT [KBps]**	**Mean NT [KBps]**	**Max NT [KBps]**	**Mean RTT [s]**	**Total counts**
S-0	677.6	1190.9	1365.9	0.439	5097
S-1	337.6	1157.2	1263.5	0.479	3284
S-2	549.6	1242.7	1352.8	0.444	4867
S-3	108.6	1252.2	1545.0	0.393	4219
S-4	75.3	1070.8	1284.2	0.490	5033
S-5	55.9	1199.0	1413.3	0.418	3484
S-6	304.3	1467.6	1616.3	0.370	4941

6.4.2 Input Data Sets

In order to evaluate the predictors, input data sets were collected in the measurement environment over a period of 48 hours. However, there is a traffic shaper at each node on PlanetLab. When the transmitted network traffic is more than the boundary volume, the shaper reduces the network traffic by force. In our measurements, the shaper reduced the traffic after 16 or 17 hours. Thus, actual throughput was only gathered for 57600 or 61200 [s] per day. Input data sets for the evaluation and simulation are shown in Table 6.2. The statistics of input sets are similar to those of the learning data sets. Next, the input data set is transformed into time-series data to reflect the throughput fluctuation for the simulation. Therefore, the time length of the input sets is approximately 110,000 or 120,000 [s]. The time-series data at site S-4 is shown in Figure 6.2. Changes in the actual throughput were found over the time.

TABLE 6.2: Characteristics of input data sets.

Site	Min NT [KBps]	Mean NT [KBps]	Max NT [KBps]	Total counts	Time length [s]
S-0	729.4	1175.4	1361.2	1129	115200
S-1	701.6	1100.7	1252.2	1095	119400
S-2	737.8	1291.6	1351.6	1057	114900
S-3	217.5	1256.7	1551.4	914	120400
S-4	680.5	1101.5	1284.2	1010	117600
S-5	373.2	1212.6	1406.7	1082	117900
S-6	583.3	1423.9	1617.1	928	120900

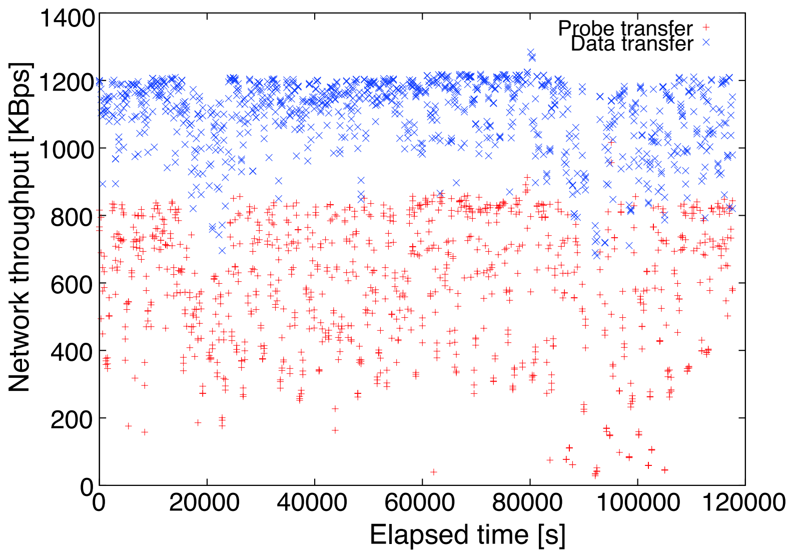
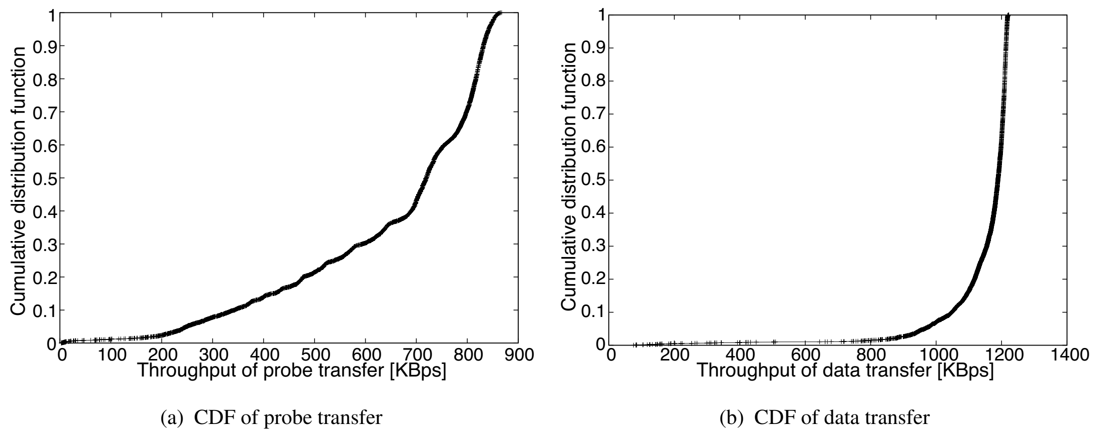


FIGURE 6.2: Time-series data of connection pair at site S-4.

6.5 Throughput Prediction Methods

6.5.1 Previous Throughput Predictor (CDF-based Predictor)

While other predictors require multiple network metrics, it is possible to build the CDF-based predictor [17] using only the connection pair. The CDF-based predictor for site S-4 is shown in Figure 6.3. The CDF curve shape of probe transfer was not similar to that of the data transfer owing to the actual throughput noise. The CDF curve shape in the other cases is similar to the above case.



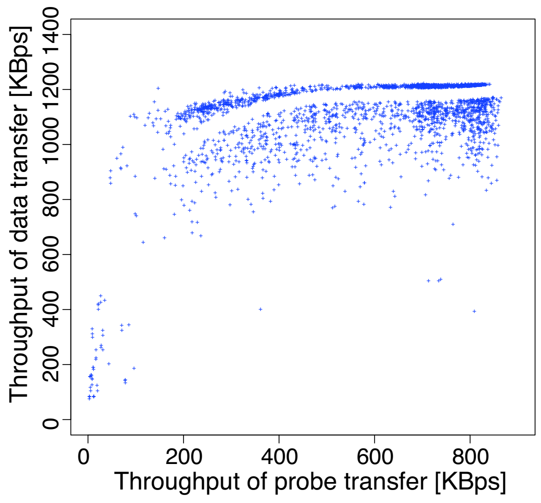
(a) CDF of probe transfer (b) CDF of data transfer

FIGURE 6.3: CDF-based predictor at site S-4.

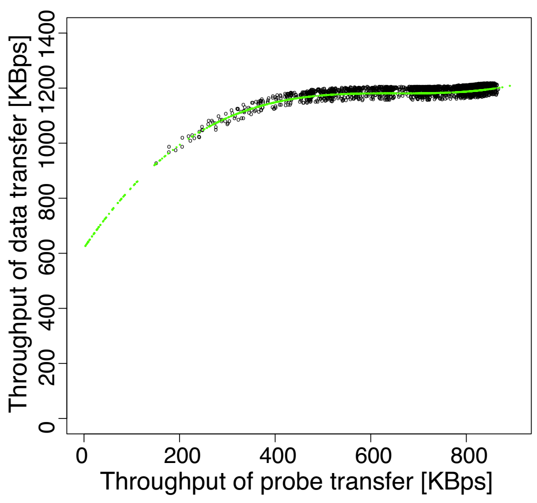
6.5.2 Proposed Throughput Predictor (SVR-based Predictor)

Before presenting the SVR-based predictor, we briefly present support vector regression (SVR) [83] that described in Chapter 5. It is a version of a support vector machine (SVM) used for regression. The concept of SVR is to maximize margins. There are two types of SVR: ϵ-SVR [83] and ν-SVR [84]. While ϵ-SVR calculates the distance of data points on a tube in order to determine the shape of tube, ν-SVR instead fixes a parameter, ν, which bounds the fraction of points lying outside the tube. Thus, there is no distance calculation on the outside tube and the appropriate tube shape can be determined using ν-SVR, even if there are outliers such as noise. ν-SVR was used to deal with the actual throughput noise in our predictor. Moreover, the kernel trick [85] can be applied in SVR without ever having to explicitly compute the mapping. The polynomial kernel was also used to consider non-linear characteristics. To summarize, our predictor uses ν-SVR and a three-degree polynomial kernel to consider a non-linear and continuous monotonic function.

The SVR-based predictor at site S-4 is shown in Figure 6.4. The SVR predictor reflects the characteristics of a non-linear and continuous monotonic function. The other cases are similar to the above case.



(a) Support vectors for ν-SVR



(b) Tube of regression curve

FIGURE 6.4: SVR-based predictor at site S-4.

6.5.3 Evaluation of Prediction Results

To evaluate the prediction results, we used the relative prediction error (RPE) [72], which is defined as

$$RPE = \frac{\hat{R} - R}{min(\hat{R}, R)} \tag{6.1}$$

where $\hat{R}$ is the predicted throughput and R is the actual throughput. The RPE fraction within 10% or less is shown, which is written as

$$Fraction\ of\ RPE = \frac{\sharp\{\boldsymbol{r} | -0.1 < RPE(r) < 0.1\}}{\sharp\{\boldsymbol{r}\}} \tag{6.2}$$

where $\boldsymbol{r}$ is the input set. Again, this way is same to the way described in Chapter 5. RPE fraction within 10% or less is shown in Fig. 6.5. The RPE fraction was higher with SVR than that with CDF in the evaluation results.

The results are similar to those in Chapter 5. While the CDF-based predictor deals with all data that include noise, our SVR predictor uses only the characteristic features in the training set. This explains its better prediction results. The regression curve of the SVR-based predictor at S-4 (Figure 6.6) was also closer to the input data set in comparison to that of the CDF-based predictor. In particular, the prediction results at the CDF-based predictor are affected by noise when the probe throughput is low. To summarize, similar prediction results are shown although different nodes were used for the measurements and predictors.

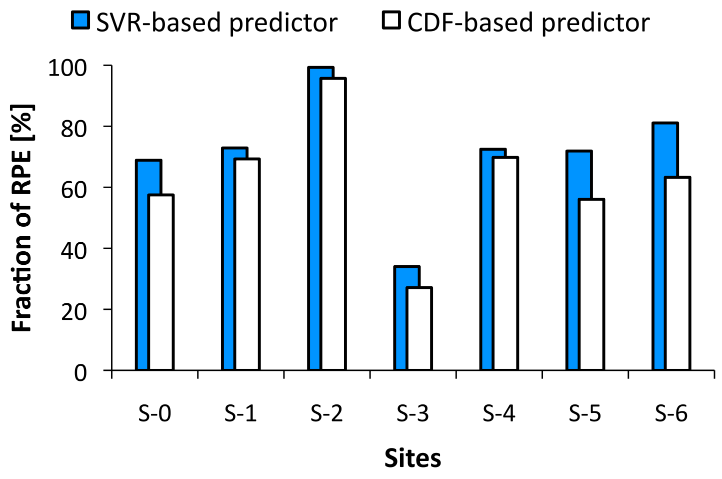


FIGURE 6.5: Fraction of RPE within 10% or less.

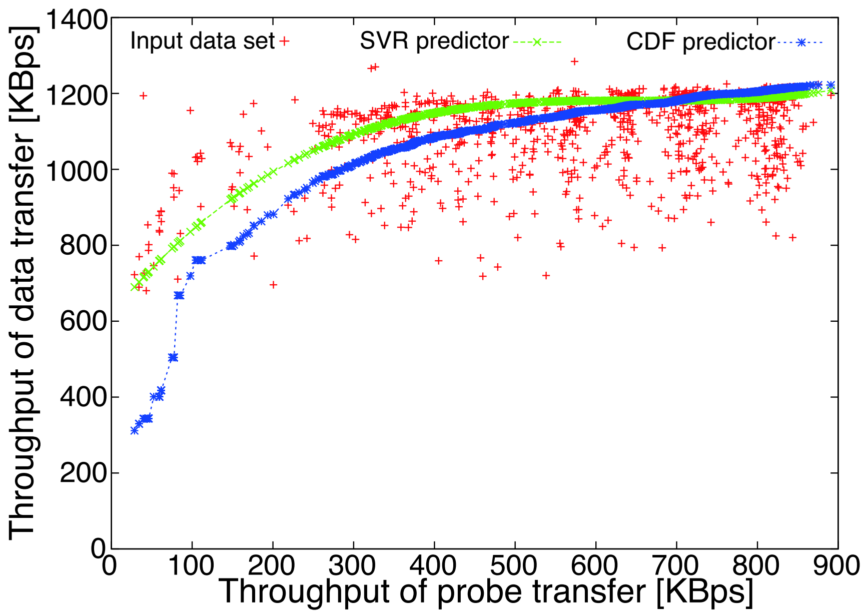


FIGURE 6.6: Prediction results at S-4.

6.6 Simulation Method

In this section, we introduce our simulation model and scenario to clarify how throughput prediction can affect task scheduler performance. We first explain a simulation model for task scheduling. Second, we present task insert periods for changing queue states. Third, we show the task schedulers for performance evaluation. Finally, we describe simulation scenario for experiments.

6.6.1 Model

The Internet has been used to share distributed resources and meta-schedulers have been used to efficiently assign tasks to resources. However, unlike computational resources, the availability of network resources on the Internet is very unstable and difficult to manage. The throughput prediction can enhance such schedulers, but it is unclear that the predictor should always guarantee the improved scheduler performance of given tasks.

Simulation with real traces can yield more realistic results than that with artificial traces. We use the input data sets as the real traces of throughput fluctuation. Again, we transform the input data sets into time-series data, which changed over time. Next, we assume that each of the sites, S-0, S-1, S-2, S- 3, S-4, S-5, and S-6, denotes a grid site. There is a local scheduler at the site for managing computational resources such as CPU, memory, and storage. For execution time, we assume that each site has the same computing power, the computational resources are integrated, and it takes 1 [s] to execute 1 MB. For example, we assume that it takes 1024 [s] to execute 1 GB on the site. The central site denotes the meta-scheduler, which has a queue that receives tasks from users, and a task is assigned to the local scheduler. There is no preemption at any of the queues; thus, the assignment is performed on a first come, first served (FCFS) basis. Our simulation model is shown in Figure 6.7.

To reflect data-intensive tasks, we assume that a task denotes a meta-task, and thus it cannot be divided. And also we assume that the task has data, thus the data should be fully transmitted to the site to process the task. In eScience, the amount of data for each experiment on the LHC [94] was approximately 1 GB per second and that on SETI@home [116] was approximately 500 MB. With reference to these sizes as well as the movement of data on the Internet, we select the following sizes. The minimum

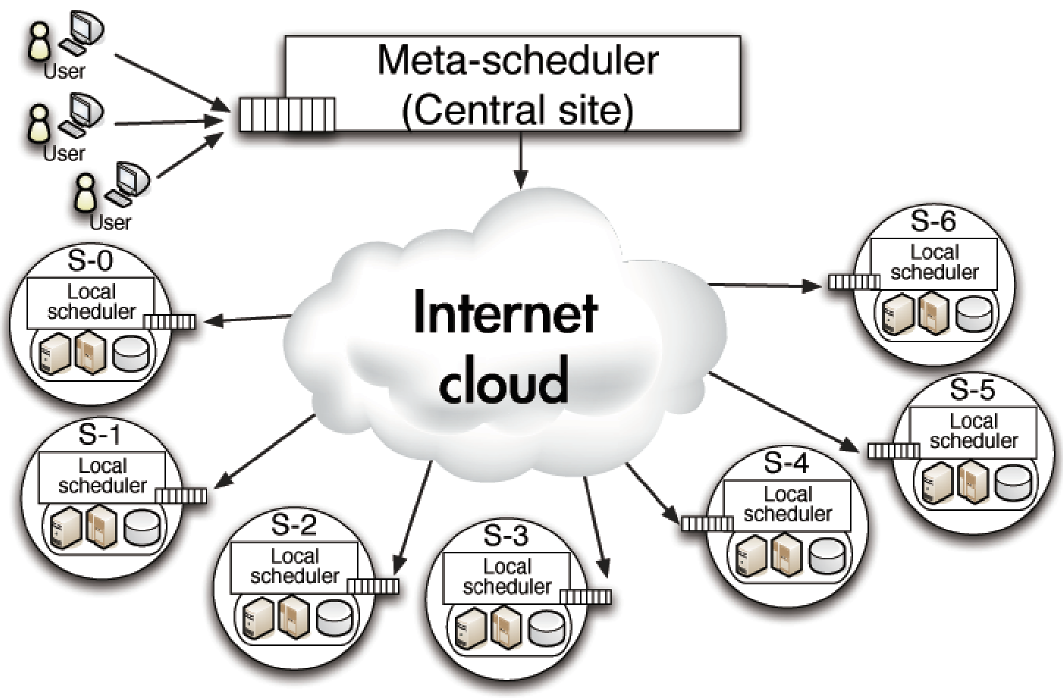


FIGURE 6.7: Simulation model.

size is 1 GB, the maximum size is 9 GB, and the data size is increased per gigabyte. We execute the simulation experiments per a task set that includes the tasks. There are four types of task sets. Although the number of tasks at the task sets was increased to 100 tasks, the results were similar after task set (IV). The task sets are summarized in Table 6.3. The data size in the task is randomly selected and the number of tasks is based on the number of sites.

TABLE 6.3: Number of tasks at task sets for simulation.

Set (I)	**Set (II)**	**Set (III)**	**Set (IV)**
7	14	21	28

6.6.2 Task Insert Periods

To change the queue state on the local scheduler on the grid site, we prepare the following task insert periods of online mode:

- ***Busy state*** limits the time slot to 300 or 600 [s]. Again, the minimum time slot is 300 [s]. Pre-scheduled tasks are presented in the local scheduler to keep the insert period short.

- ***Idle state*** considers the time slot as 3600 [s]. The insert period is larger than the busy state. The pre-scheduled tasks almost never remain in the local scheduler.

If the start time is fixed as zero or a certain time, similar results will be expected and the full time-series data cannot be used. While a start time ranging from 0 to 99000 [s] is randomly selected in the busy state, a start time ranging from 0 to 45000 [s] is randomly selected in the idle state in order to use all of the data.

6.6.3 Meta-schedulers

In our simulation, the basic scheduling approach involves assigning the task to the site having the minimum expected completion time (MCT) [117], which is defined as

$$MCT(t_i, s_j) = min_{0 \leq j < N}(CT(t_i, s_j) + ETP(s_j)) \tag{6.3}$$

where s is the site, t is the task at the task set. i is the index of task and j is the index of site. N is the number of the grid sites. This time N is 7. Moreover, $CT(t_i, s_j)$ is the expected completion time when the task t_i is assigned to the site s_j, and the calculation of $CT(t_i, s_j)$ depends on the scheduler. $ETP(s_j)$ is the execution time of the pre-assigned tasks on the site s_j. If task scheduling is started, $ETP(s_j)$ will be zero on site s_j.

The following schedulers are used for task scheduling:

- ***COM*** Only the computational resource is considered when assigning tasks. If there are pre-scheduled tasks at the sites, this scheduler calculates the execution time at each site and the task is allocated to the site that has the minimum execution time. Conversely, if there are no tasks at the sites, the task is sequentially assigned from S-0 to the last site. For example, the first task would be allocated to site S-0. This is defined as

  $$CT(t_i, s_j) \quad = \quad ET(t_i, s_j) \tag{6.4}$$

  where $ET(t_i, s_j)$ is the execution time of task t_i on site s_j.

- ***NT(SVR)*** The computational resource and the predicted throughput using the SVR predictor are considered when assigning tasks. Thus, this scheduler calculates combinations of the execution time and the predicted transfer time using the SVR predictor, and then selects the minimum combination. This is defined as

$$CT(t_i, s_j) = ET(t_i, s_j) + DT_{svr}(t_i, s_j) \tag{6.5}$$

where $DT_{svr}(t_i, s_j)$ is the predicted transfer time of task t_i found by using the SVR predictor for site s_j.

- ***NT(CDF)*** This scheduler is the same as NT(SVR), but it uses the predicted throughput that is obtained by using the CDF predictor. We prepared this scheduler to clarify the performance using different prediction results. This is defined as

$$CT(t_i, s_j) = ET(t_i, s_j) + DT_{cdf}(t_i, s_j) \tag{6.6}$$

where $DT_{cdf}(t_i, s_j)$ is the predicted transfer time of task t_i estimated by using the CDF predictor for site s_j.

- ***NT(Ideal)*** This scheduler is the same as NT(SVR), but it uses the perfectly predicted throughput. Thus, there is no error between the predicted and actual throughput. This scheduler is used to compare the above schedulers with the throughput prediction. It is defined as

$$CT(t_i, s_j) = ET(t_i, s_j) + DT_{ideal}(t_i, s_j) \tag{6.7}$$

where $DT_{ideal}(t_i, s_j)$ is the actual transfer time of task t_i for site s_j.

6.6.4 Scenario

The simulation experiments were executed 10 times per the task set in order to clarify the simulation results. The simulation scenario is as follows:

1. The start time of scheduling is randomly chosen from the range of time in the time-series data of real traces.
2. Each of the schedulers calculates the expected completion time and selects the grid site that has the minimum expected completion time (MCT) when the task is inserted in the scheduler.
3. We assume that the data is promptly transmitted to the selected site.

4. After arriving the entire data, the task will wait in the local scheduler until the pre-scheduled tasks are completed if there are pre-scheduled tasks at the site. Conversely, if there are no tasks in the local scheduler, the task will be promptly processed.

5. Steps 2 to 4 are repeated until the all tasks are assigned.

To summarize, we assume that the task that has data (1-9 GB) is transmitted to the selected grid site and processed at the site. For our simulation method, we design and implement the calculation of processing time on the schedulers.

6.7 Evaluation of Simulation Results

In this section, we present evaluation results of the simulation to compare the performance of each task scheduler. Furthermore, we observe the operation of our simulation method and the effect of throughput prediction on the performance of our method. We first introduce the evaluation criteria. Next, we compare the performances of the simulation results. Finally, we clarify a major cause of decreased performance that is unexpectedly observed during the simulation.

6.7.1 Scheduling Performance Rate (SPR)

To evaluate the simulation results, we determine the processing time, which is a combination of the actual transfer time and the execution time for given tasks, and use the scheduling performance rate (SPR), which is defined as

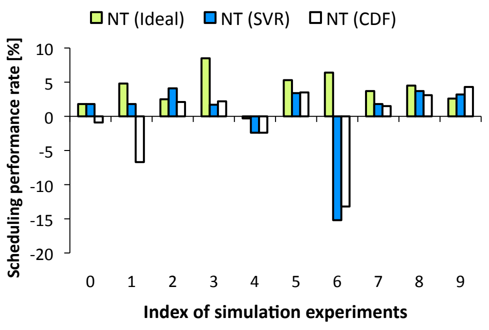
$$SPR = \frac{T_{com} - T_{nt}}{T_{com}} * 100 \tag{6.8}$$

where T_{com} is the processing time of COM and T_{nt} is the processing time of NT(SVR), NT(CDF), or NT(Ideal). A positive SPR value means that task scheduling with throughput prediction achieved better performance than task scheduling without any predictions. Thus, the SPR is a differential rate between the processing times.

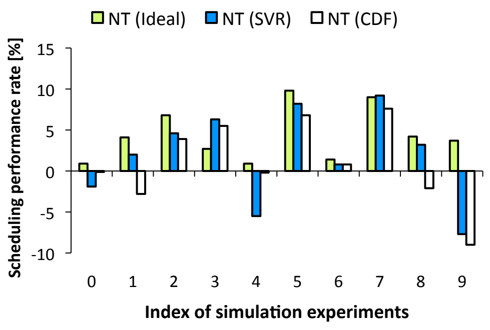
6.7.2 Simulation Results

Figures 6.8 and 6.9 show SPR values of simulation results, consisting of 10 trials on each task set. Thus, we focus on the evaluation of the task set.

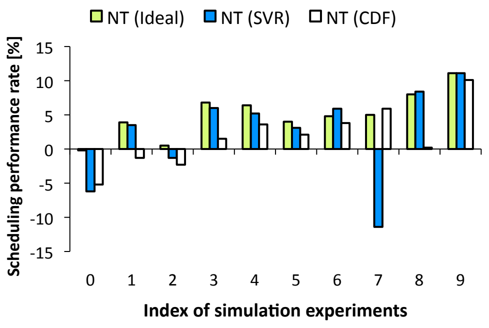
Most of the results showed improved performance; thus, the processing time is reduced as compared to that without the throughput prediction. At index 0 for the busy state (Figure 6.9(b)), the overall processing time at COM is 150088.4 [s] and that at NT(SVR) is 136354.9 [s]. Then, the SPR at NT(SVR) is approximately 9.2%, which is the same as that at NT(Ideal). On the other hand, some results showed decreased performance. Two causes of the decreased performance are considered: one is improper site selection based on MCT, and the other is imprecise prediction. The result of NT(SVR) at index 2 for the busy state is attributable to site selection because the SPR at NT(SVR) is close to that at NT(Ideal). Because of the large number of tasks in the queue, only a few idle sites have low throughput. The above situation results in the negative values at NT(Ideal). Others are similar to the above result. Next, the result at index 9 for the idle state is attributable to imprecise prediction because the SPR at NT(Ideal) is increased. We will discuss this case later. Our simulation method, which evaluates the effect of throughput prediction on the scheduler, showed improved performance with real traces.



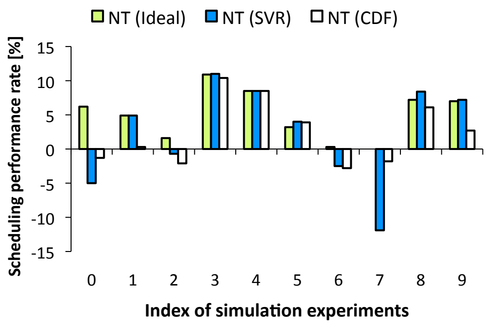
(a) Task set (I)



(b) Task set (II)

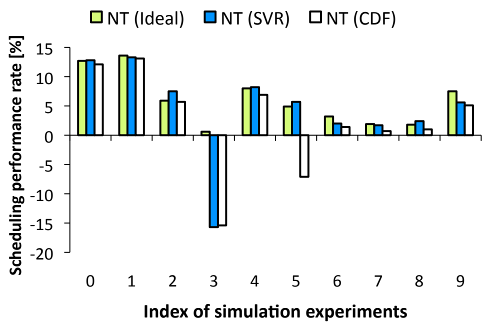


(c) Task set (III)

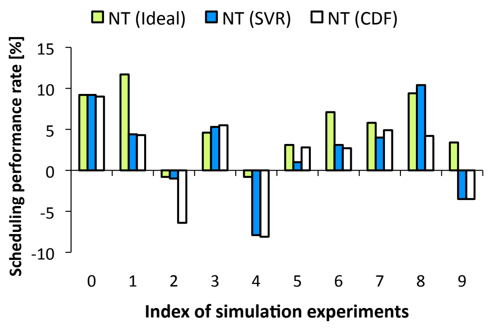


(d) Task set (IV)

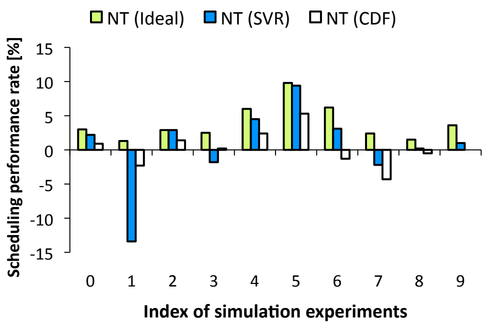
FIGURE 6.8: SPR of simulation results in idle state.



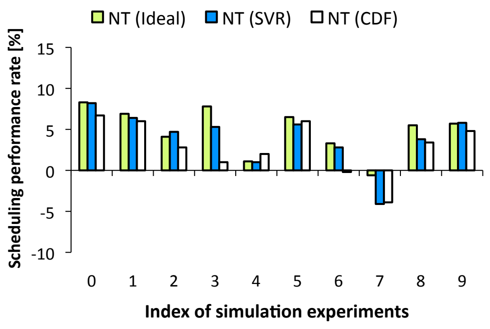
(a) Task set (I)



(b) Task set (II)



(c) Task set (III)



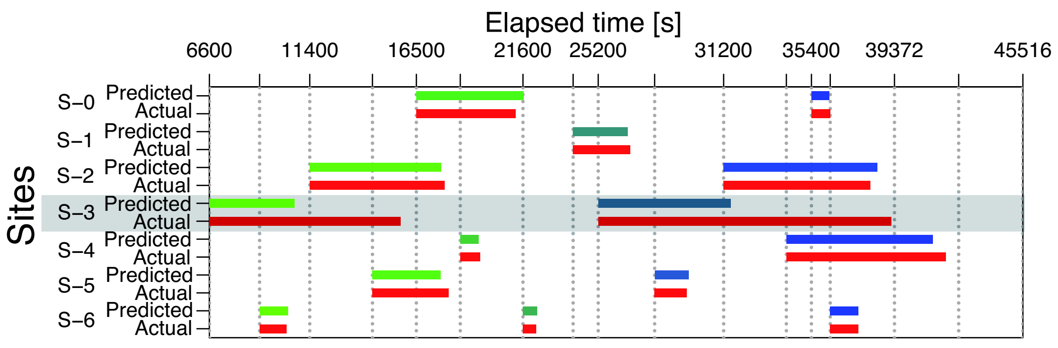
(d) Task set (IV)

FIGURE 6.9: SPR of simulation results in busy state.

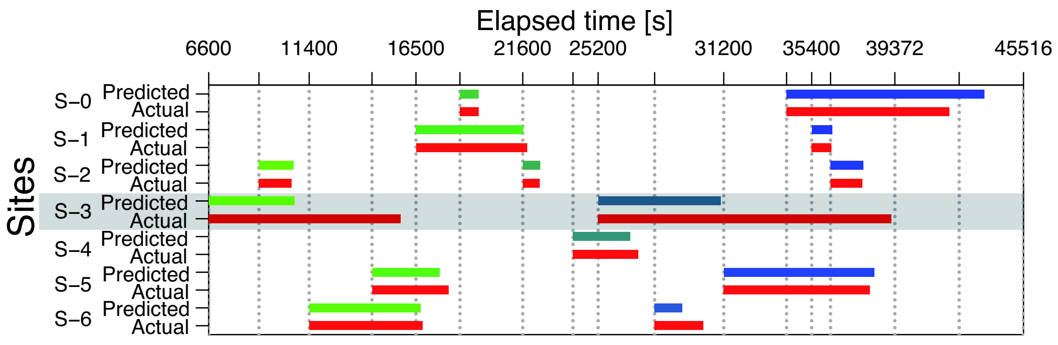
6.7.3 Cause of Decreased Scheduling Performance

To clarify the cause of decreased performance due to imprecise prediction in detail, we investigate the cases at index 9 in Figure 6.8(b) and index 4 in Figure 6.9(b). Figures 6.10 and 6.11 illustrate Gantt charts, each of which shows a comparison between the actual transfer time and predicted time for each task. In the figures, the task color is changed from green to blue for the predicted time. The first task is depicted in green and last task, in blue. The actual task is shown in red. The length of each bar corresponds to the transfer time. The larger the data size of a given task, the longer is the corresponding bar.

At index 9 for the idle state (Figure 6.10), the SPR values at NT(SVR) and NT(CDF) are -7.7% and -9.0%, respectively. The actual time is similar to the predicted time, except for site S-3 because the case at site S-3 is different from the cases at other sites. Even though site S-3 is only selected twice, the predicted and actual time at NT(SVR) are 10410.7 [s] and 23164.4 [s], respectively. The imprecise prediction is a major cause of the decreased performance. The case at NT(CDF) is also similar to that at NT(SVR). For the busy state at index 4 (Figure 6.11), the cause is the same as in the above case. These cases are enough to show that the predictors should not always guarantees the improved scheduler performance. And also the higher accuracy should not always guarantee the improved performance. Similar cases are also found for the other task sets. This implies that the performance degradation can be mitigated by detecting and re-scheduling such incorrectly scheduled tasks. However, we do not consider this mitigation for now. Moreover, we find that the multiple data transfers are overlapped in the busy state. It implies that the transfer time may become much longer. To consider this situation, we generated three connection pairs simultaneously in our measurements. Moreover, aggregate throughput is not saturate on PlanetLab when the number of connections is below 5 [42]. In future, we intend to explore parallel transfers on the Internet and include a model of the parallel transfers to consider the overlapped transfers.

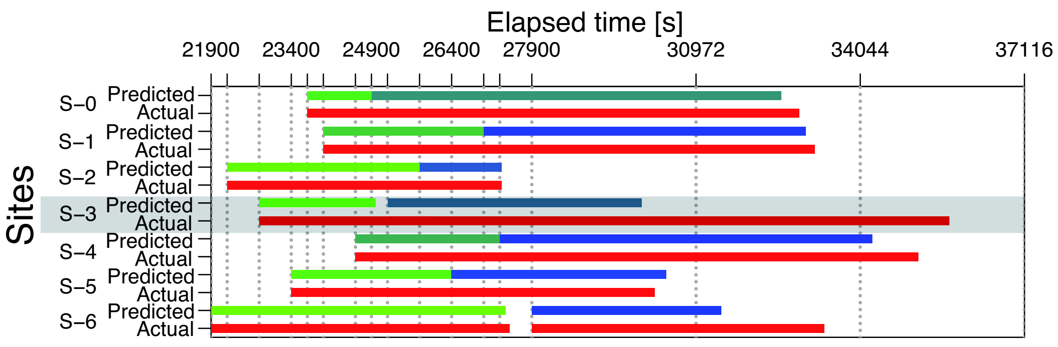


(a) Transfer time of NT (SVR) at index 9

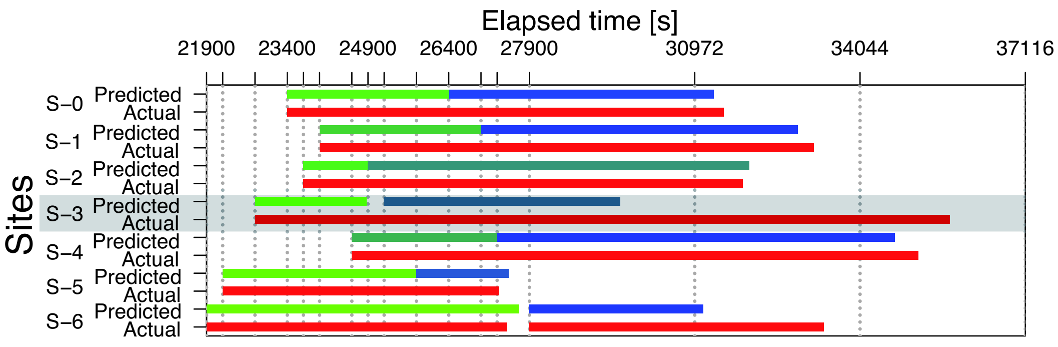


(b) Transfer time of NT (CDF) at index 9

FIGURE 6.10: Gantt chart of idle state for task set (II).



(a) Transfer time of NT (SVR) at index 4



(b) Transfer time of NT (CDF) at index 4

FIGURE 6.11: Gantt chart of busy state for task set (II).

6.7.4 SPR of Entire Result Set

Here, we discuss on the entire simulation result set to clarify the contribution of predictors to overall performance and verify whether the overall NT(SVR) performance is improved in comparison to that of NT(CDF). We prepare a data set composed of the 80 simulation results at each of the schedulers.

In order to clarify the expected scheduler performance, we calculate the expected values of SPR at each of the schedulers. The expected value of SPR at NT(Ideal) is 4.85%. It is an upper value of SPR, thus any scheduler cannot overcome this upper value under the simulation method we used. In other words, we can expect the improved scheduler performance if the expected value is close to the upper value. The expected values are given in Table 6.4. The expected value at NT(SVR) is close to that at NT(Ideal). It is obvious to expect the improved scheduler performance when the SVR-based predictor is used.

TABLE 6.4: Expected values of SPR for entire result set.

Scheduler	Expected values of SPR [%]
NT (Ideal)	4.85
NT (SVR)	3.73
NT (CDF)	2.53

To investigate the performance of the entire result set, we calculate the CDF of the SPR for each scheduler. The CDF of SPR is depicted in Figure 6.12. For SPR values larger than -2.4%, the curve shape of NT(SVR) is close to that of NT(Ideal) in comparison to NT(CDF). When the CDF value is 0.5, the SPR at NT(SVR) is 3.1% and that at NT(CDF) is 1.5%. To investigate the SPR fraction with the decreased performance, we select the value at which SPR is 0.0%. The SPR fraction less than 0.0% at NT(SVR) is 23.8% and that at NT(CDF) is 33.8%. The SPR fraction at NT(SVR) is smaller than that at NT(CDF). However, the results show that the predictors should not always guarantee the improved scheduler performance of the given task set. Next, the SPR statistics for the entire result set are summarized in Table 6.5. The mean and median SPR values at NT(SVR) are higher than those at NT(CDF) and close to NT(Ideal). In particular, the maximum SPR at NT(SVR) is 13.3%. Thus, the processing time is reduced by up to 13.3% as compared to the scheduler without any throughput predictions. There is a significant performance improvement in the overall results when we use the predicted throughput using the SVR-based predictor under our simulation method.

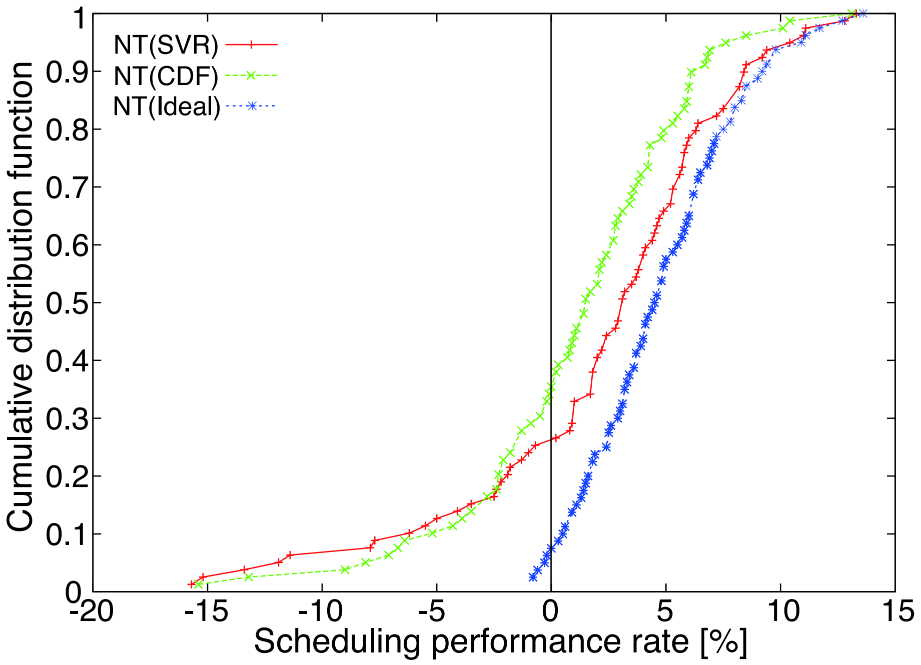


FIGURE 6.12: CDF of SPR for entire result set.

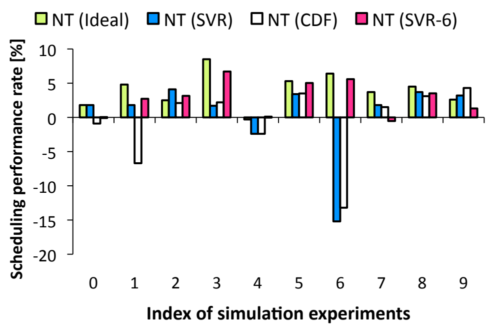
TABLE 6.5: SPR statistics for entire result set.

Scheduler	**Minimum SPR [%]**	**Median SPR [%]**	**Mean SPR [%]**	**Maximum SPR [%]**
NT (Ideal)	-0.8	4.6	4.8	13.6
NT (SVR)	-15.7	3.2	2.3	13.3
NT (CDF)	-15.4	1.6	1.3	13.1

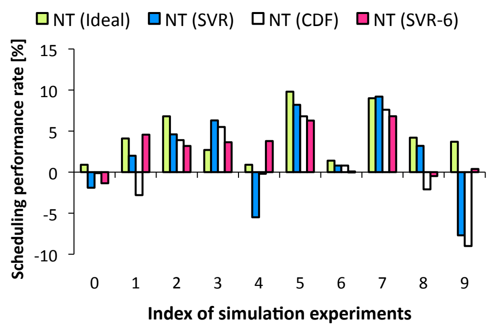
6.7.5 Simulation Results without Site S-3

When there is no large prediction error, we should use all of sites and an accurate predictor will have the largest SPR value. Normally, this case is predictable. But, a situation with large prediction error is different from the above situation. Thus, we should consider whether to include a site with large prediction error or to exclude it for the task scheduling. Again, large prediction error at site S-3 is a major cause of negative SPR values. We here explore additional simulations using the SVR-based predictor without site S-3 (as indicated NT(SVR-6)) to investigate the effect of large prediction error.

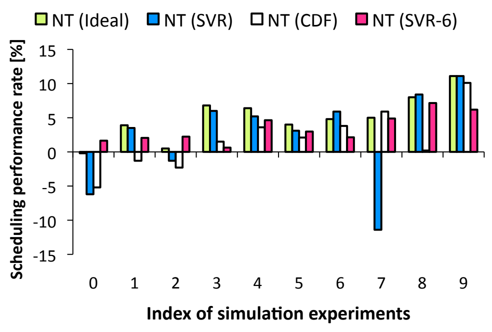
Simulation results at NT(SVR-6) are shown in Figures 6.13 and 6.14. At results for index 5 in the idle state (Figure 6.13(b)), NT(Ideal) has the largest SPR value and NT(SVR-6) has the smallest SPR value. In the above results, there is no large prediction error. However, results with large prediction error are different from the results in Section 6.7.3. At results for index 9 in the idle state (Figure 6.13(b)) and index 9 in the busy state (Figure 6.14(b)), the result at NT(SVR-6) has the positive SPR value while those at NT(SVR) and NT(CDF) have the negative SPR values. Most results with large prediction error are similar to the above cases. It is enough to show the effect of large prediction error. If a site has sufficient the computational resource and there is large prediction error or an abrupt change in network state, we should carefully consider the use of that site.



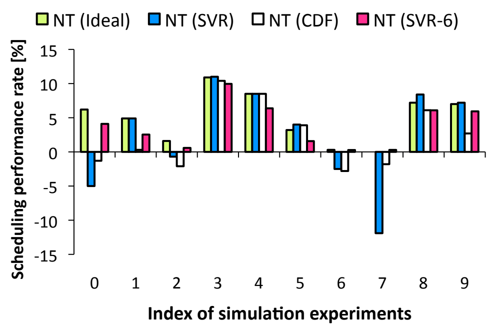
(a) Task set (I)



(b) Task set (II)

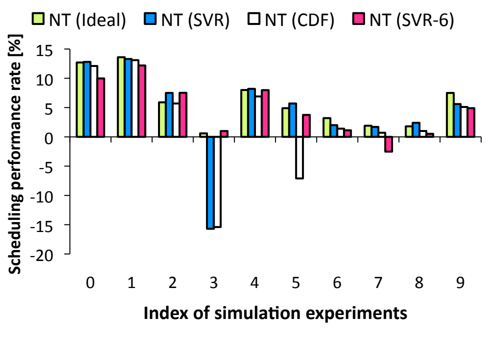


(c) Task set (III)

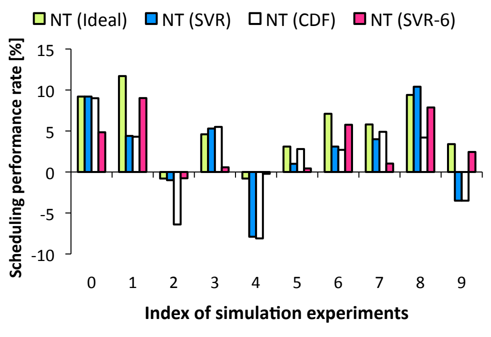


(d) Task set (IV)

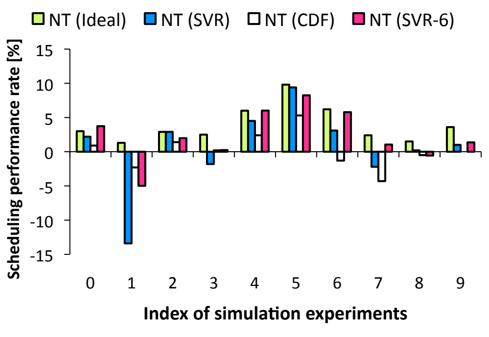
FIGURE 6.13: SPR of simulation results in idle state without site S-3.



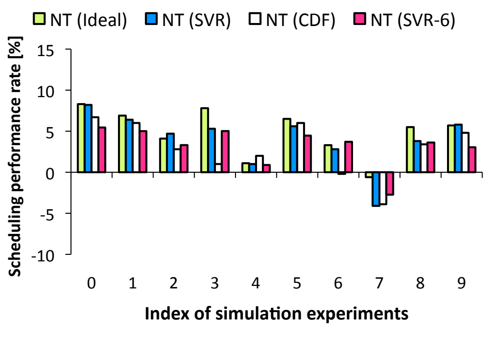
(a) Task set (I)



(b) Task set (II)



(c) Task set (III)



(d) Task set (IV)

FIGURE 6.14: SPR of simulation results in busy state without site S-3.

6.8 Implications

In this section, we analyze the entire result set through statistical analysis for a better understanding of the simulation results. We calculate a significance level through sign test to determine whether the simulation results at NT(SVR) are valid. Then, we investigate the relationship between performance and predictability through the correlation coefficient.

The sign test is a non-parametric test, thus there are very few assumptions about the nature of distributions. It is used to test the null hypothesis that the median of a distribution is equal to some value. To show the validity of the simulation results at NT(SVR), we calculate a significance level through the sign test. The counts of winning and losing at NT(SVR) are 59 and 21, respectively. The significance level is 0.000012, which is less than 0.01. This result sufficiently confirms the validity of the simulation results at NT(SVR).

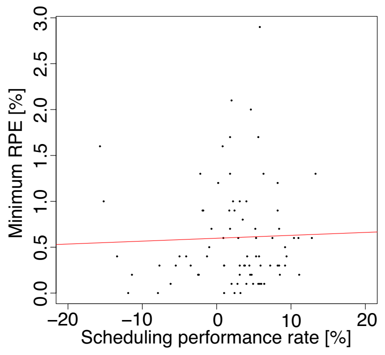
Most of our simulation results have improved performances using the predicted throughput, although a few results have decreased performances resulting from imprecise prediction. In other words, the prediction error increases when performance decreases without the improper site selection. A negative relationship between performance and predictability is implied by the simulation results.

To determine an appropriate parameter for the relationship, we calculate a correlation coefficient between the SPR and the statistics of RPE. The correlation coefficient is given in Table 6.6. The scatter plots of correlation coefficient at NT(SVR) and NT(CDF) are shown in Figures 6.15 and 6.16. In the scatter plots between SPR and the minimum RPE (Figures 6.15(a) and 6.16(a)), the correlation values are close to zero and there is no linear correlation. The correlation values between SPR and the median RPE (Figures 6.15(b) and 6.16(b)) are also similar to the above results. Conversely, there is a strong negative correlation between SPR and the mean RPE (Figures 6.15(c) and 6.16(c)). In particular, the result at NT(SVR) is higher than that of NT(CDF). The similar results are shown between SPR and the maximum RPE (Figures 6.15(d) and 6.16(d)).

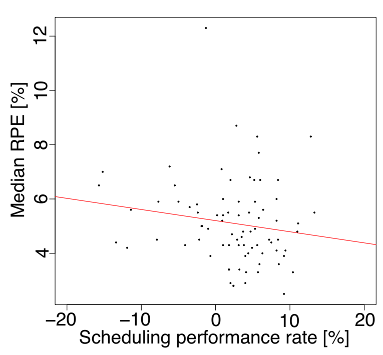
TABLE 6.6: Correlation coefficient between SPR and statistics of RPE.

Scheduler	**Minimum RPE**	**Median RPE**	**Mean RPE**	**Maximum RPE**
NT (SVR)	0.03	-0.15	-0.75	-0.72
NT (CDF)	-0.03	-0.10	-0.66	-0.54

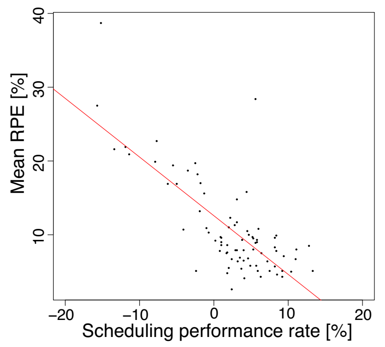
These evaluations and analysis can be summarized as follows. Under our simulation method based on real traces of throughput fluctuation, it is clear that the predicted throughput using the SVR-based predictor improves performance. However, the schedulers should not always improve the scheduler performance. And also the higher accuracy should not always guarantee the improved performance. Next, the CDF curve at NT(SVR) is close to that at the NT(Ideal) in comparison to NT(CDF), and the expected value of SPR at NT(SVR) is also larger than that at NT(CDF). Then, the significance level is observed to be smaller than 0.01 through the sign test, which sufficiently confirms that the simulation results at NT(SVR) are valid. Finally, there is the strong negative correlation between SPR and the mean RPE.



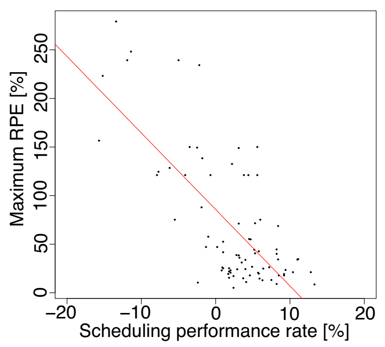
(a) Minimum RPE



(b) Median RPE

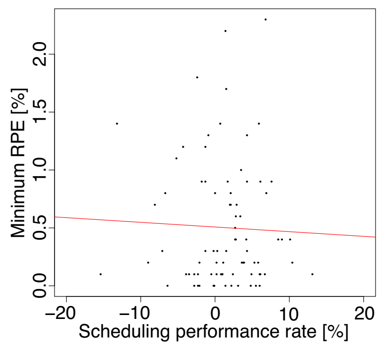


(c) Mean RPE

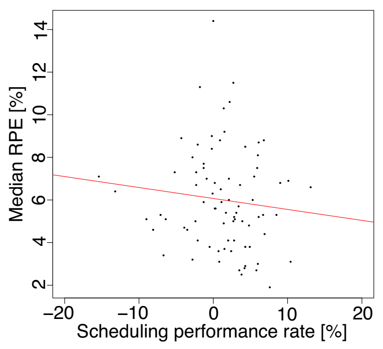


(d) Maximum RPE

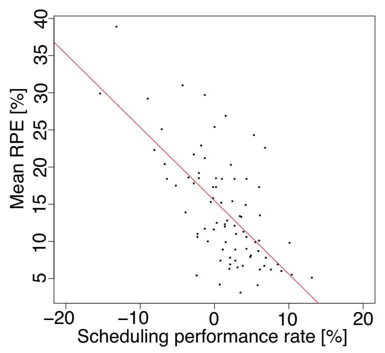
FIGURE 6.15: Scatter plots of correlation coefficient at NT(SVR).



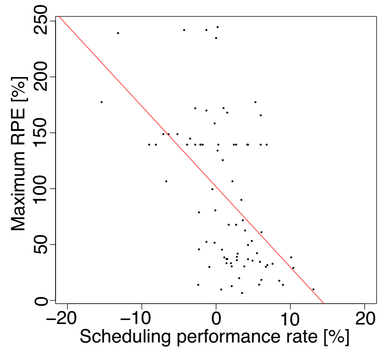
(a) Minimum RPE



(b) Median RPE



(c) Mean RPE



(d) Maximum RPE

FIGURE 6.16: Scatter plots of correlation coefficient at NT(CDF).

6.9 Conclusion

In this chapter, we show the simulation where the proposed SVR-based predictor improves the performance of computation. Our approach was based on simulations using real traces of throughput fluctuation on the Internet. We developed a simulation method that evaluates the impact of throughput prediction on the scheduler performance using real traces. Most results show improved performance using the predicted throughput under our simulation method. In particular, the scheduler using the SVR-based predictor leads to an overall processing time reduction of up to 13.3% compared to the scheduler without any throughput predictions. The SVR-based predictor, which is a more precise predictor than the CDF-based, showed significant performance improvement in the overall results. Moreover, the expected value of SPR using the SVR-based predictor is closer than that of ideal case. However, the schedulers using the throughput predictors should not always improve the scheduler performance. Although precise prediction results increase the scheduler performance, only a few of imprecise prediction results drastically decrease the entire scheduler performance. Finally, a better understanding of the results was achieved through statistical analysis. We found that the result using the SVR-based predictor was appropriately representative of a strong negative correlation between predictability and performance.

Chapter 7

Conclusion and Future Directions

7.1 Overview

The previous chapters have discussed the research issues related to Internet traffic characteristics, the accurate prediction model, and the contributions of prediction results. The Internet traffic characteristics are obviously presented with the virtualization technology, precise predictions have been enabled in comparison with the previous prediction method, and the contributions of prediction have been clarified through the metascheduler simulation.

We divide this chapter into three sections: Section 7.2 summarizes the whole thesis and presents a comprehensive review of the issues and approaches that have been taken throughout the thesis. Section 7.3 describes conclusion with the contributions and identifies their significances. Section 7.4 discusses the future research directions. Finally, we conclude the thesis with a closing statement.

7.2 Summary

In this thesis, we presented a throughput prediction method based on the connection pair that uses ν-SVR and polynomial kernel to deal with prediction models represented as a non-linear and continuous monotonic function. Because the SVR-based predictor uses only characteristic features in the training set, the prediction results of the SVR-based predictor are more accurate and robust than the CDF-based predictor for the same data sets. The research questions have been discussed and answered through various experiments and analysis results.

In Chapter 2, we investigated and discussed many issues of Internet traffic characteristics when the virtualization technology is used on the Internet. In measuring the throughput, we showed that oversize packet spacing, which can be caused by CPU scheduling latency, is a major cause of throughput instability on the virtualized testbed even when no significant changes occur in the well-known network metrics. Particularly, some of the packet spacings were larger than RTT, and these oversize packet spacings were unusual anomalies on virtualized network environment. Moreover, we determined the anomalous case caused by the anomalies. It is throughput instability despite of stable network state, which can be observed through RTT, packet loss rate, and advertised window. The condition for the judgment of the anomalous case is the impact of packet spacing. We should carefully review measurement results obtained under such the anomalous cases. In monitoring resources, we showed that the anomalous cases are occurred when the CPU utilization is high and the CPU availability is low. Our empirical approach, which observes criteria provided by system during the throughput measurement and analyzes the criteria statistically, enables the anomalous cases to be identified on the virtualized testbed. We found that the CPU availability is an important criterion for estimating the anomalous case. On the other hand, we explored occurrences of the anomalies on Amazon EC2 to investigate on different virtualized environment. We measured the throughput for the connection pair between the micro instance on Amazon EC2 and a native node on our university, and observed correct resource state by using Amazon CloudWatch [52]. Analysis results showed that very few of packet spacings are the anomalies, and it is hard to determine the major cause of throughput instability. Our results are different from the results shown by Wang et al. [50]. Moreover, packet loss was occurred continuously during the throughput measurements although RTT was very similar to mean RTT using ping. It was the major cause of throughput instability. In data center, various types of many flows would co-exist, and they would affect their communication performance. The CPU utilization was less than 20% during the measurements. The CPUs were not busy, and the CPUs were fully allocated to the instances. Thus, there was no CPU scheduling latency. In data center, the CPU utilization would be low due to resource policy and the structure of data center.

In Chapter 3, we focused on a nature of resource state to make up for a weak point in the CPU availability. We applied PCA to a matrix gathered from the resource state to establish the criteria for anomaly estimation. Analysis results showed that the top two principal components account for 84% of the original data set and can describe the original resource state. Component loadings and a scatter plot of the first and second component scores provided a simple but descriptive enough view of the resource state for estimating the anomalous case. The first component can be interpreted as workloads, and the second one as lack of resources, leading to the anomalous case. These components can be used instead of the CPU availability to estimate the anomalous case. We determined the appropriate boundaries of the components by using Bayes' approach and used them to automatically evaluate the anomalous case with an input data set gathered from other nodes. The evaluation results presented the validation of our approach.

In Chapter 4, our approach for the selection of an appropriate probe size was described before the prediction, and the validity of the selected probe size was shown. We selected the probe size through Spearman's rank correlation coefficient (ρ), thus our selection is based on a predictability of probe. We clarified that probe size obtained by using unfiltered data is not always appropriate for the connection pair. Daily changes in transfer time at this connection pair are occurred at the certain period, and a major cause of throughput fluctuation was the anomalous case. As a result, the inappropriate probe size had high predictability. Without the anomalous case, the 256-KB probe had the highest ρ. To investigate whether this probe is appropriate or not for the prediction, we generated the connection pair when the probe size was 256 KB and the data size was 16 MB on the virtualized testbed. When the probe size is used to 256 KB, actual throughput for the connection pair had a non-linear and changed monotonically. Conversely, when the probe size for the connection pair is decreased to 32 KB or 64 KB, the actual throughput was concentrated on certain areas or considerable noise occurs in the actual throughput. The selected probe size through statistical analysis is appropriate for the connection pair.

In Chapter 5, a description of the SVR-based predictor and its characteristics were provided. We first built a predictor based on an existing prediction method [17] for the same data sets as for our method to evaluate whether it produces precise prediction results. We found that the existing prediction method was unsuitable when the actual throughput was non-linearly and monotonically changed with noise. We thus proposed a throughput prediction method for precise prediction results. The proposed method uses ν-SVR and the polynomial kernel to deal with prediction models represented as a non-linear and continuous monotonic function. The prediction results of our proposed method are more accurate than those of the existing one. Furthermore, it is more robust than the existing one under an unstable network state. Next, we changed the probe size for the connection pair to investigate the predictability of the SVR-based predictor. There were no significant changes in prediction results with 512-KB probes. Conversely, there was considerable noise at the actual throughput when the probe size was decreased to 64 KB. The noise is an obstacle to finding an appropriate regression curve. Prediction results are inaccurate in comparison with the original SVR-based predictor. As a result, the 256-KB probes are appropriate for the current networks, and our SVR-based predictor is accurate, robust, and suitable for its purpose. To summarize, we focused on measurements, analysis, and modeling for precise prediction results. Our approach is presented through Chapters 2 and 5.

In Chapter 6, we focused on how the predictors affect meta-scheduler performance quantitatively. Our study was based on simulations using real traces of throughput fluctuation on the Internet. We developed a simulation method that evaluates the impact of throughput prediction on the scheduler performance using real traces. Most results show improved performance using the predicted throughput under our simulation method. In particular, the scheduler using the SVR-based predictor leads to an overall processing time reduction of up to 13.3% compared to the scheduler without any throughput predictions. The SVR-based predictor, which is a more precise predictor than the CDF-based, showed significant performance improvement in the overall results. Moreover, the expected value of scheduling performance rate (SPR) using the SVR-based predictor is closer than that of ideal case. However, the schedulers using the throughput predictors should not always improve the scheduler performance. Although precise prediction results increase the scheduler performance, only a few of imprecise prediction results can drastically decrease the entire scheduler performance. Next, a better understanding of the results was achieved through statistical analysis. We found that the result using the SVR-based predictor was appropriately representative of a strong negative correlation between predictability and performance.

7.3 Contributions

In this research, we proposed the SVR-based predictor with the improved prediction results. Our approach is based on measurements, analysis, and modeling for the precise predictor. Moreover, we quantitatively clarify how the predictors can affect the overall processing time of given tasks through meta-scheduler simulation. Our results present the validation of research questions.

The main contributions of this research are the following:

- With the virtualization technology, extra effects are specifically observed on network measurement. In particular, throughput instability is occurred by oversize packet spacings although there are no significant changes in the well-known network metrics. A major cause is CPU scheduling latency among virtual platforms. If network throughput is decreased by the impact of virtualization, we should carefully review measurement results. In our experiments, a 32-KB probe had the highest predictability because changes in throughput were occurred by the impact of virtualization. When we get rid off the negative impacts, it was not the best condition. It validates research question 1 as outlined in Chapter 1 to characterize Internet traffic when the virtualization technology is used on the current Internet.
- In order to estimate the impact of virtualization, we proposed two methods: the former focuses on the CPU availability and the latter is based on a nature of resource states with principal component analysis (PCA). It is easy and simple to measure the CPU availability, but it overconsumes CPU resources and affects the performance of other tasks. Meanwhile, when we applied PCA into the resource states, the top two principal components account for 84% of the original data set and can describe the original resource state. Moreover, the first component is workloads on the node, and the second one is lack of resources, leading to oversize packet spacings. These components can be used instead of CPU availability to estimate the oversize packet spacings. This validates research question 2 that states that the estimation methods on application layer are required for the impact of virtualization.
- In the current Internet, various types of traffic, such as mice and elephant flows, co-exist, and spikes that correspond to large and abrupt throughput are occasionally caused by the elephant flows. As a result, probability distribution of traffic is unclear or close to long-tail by the above characteristics. Under the above situations, it is hard to expect linearity between probe and data transfers. Thus, a non-linear and monotonicity between probe and data transfers are expected. Moreover, a probe size should be increased in comparison with the original probe size (64 [KB]) because the scale and bandwidth of network are rapidly increased. In Chapters 4 and 5, we showed that the 64-KB probe has considerable noise and it was shrunk in certain throughput areas. Meanwhile, the non-linear characteristic and monotonicity were found with the 256-KB probe. This addresses research

question 3 that investigates the characteristics between probe and data transfers. Moreover, it partially validates research question 4.

- An appropriate probe size should have high predictability and reflect congestion window correctly. To select the appropriate probe size, we assumed that there is monotonicity between probe and data transfers. We gathered approximately 15,000 connection pairs, and evaluated through Spearman's rank correlation coefficient (ρ). It is a valid indicator because it focuses on the ranks of data set, and does not rely on probability distribution of data set. In our evaluation results, a 256-KB probe had the highest rank correlation coefficient (ρ = 0.67), and it was selected for the prediction model. Our approach is unique in the sense that the probe size is selected based on the predictability before the prediction. This addresses research question 4 about the characteristics between probe and data transfers.

- To take account into noise and non-linear characteristics, we applied ν-SVR and polynomial kernel into our data sets. For noise, ν-SVR is more suitable than ϵ-SVR because it does not calculate the distance of data points on out side. Thus, even if there is outlier, such as noise, we can determine the appropriate shape of tube. It is a different point in comparison with ϵ-SVR. Prediction results with ν-SVR are also more accurate than those with ϵ-SVR. To deal with non-linear characteristics, we used polynomial kernel of degree 3. Moreover, we performed the same experiments by varying the degree from 2 to 5 to investigate the adequacy of degree 3. Although the degree is changed for the prediction model, there are no significant differences. As a result, our approach is a reasonable choice for our prediction method. This indicates a solution of research question 5.

- To clarify the contributions of prediction results, we explored meta-scheduler simulation with real-traces of throughput. The meta-scheduler using the SVR-based predictor was observed to result in a reduction of up to 13.3% in the overall processing time compared to the meta-scheduler without any predictions. We also show that the schedulers with throughput predictors should not always guarantee the reduction of processing time. Only a few of large prediction errors can drastically affect the overall processing time. As a result, it takes more time than the scheduler without any predictions although precise prediction results reduce the processing time. To mitigate such performance degradation, detecting or re-scheduling mechanism should be considered. It answers research question 6 concerned in the contributions of prediction results.

These conclusions adequately answered the research questions described in Chapter 1. This research is unique in the sense that it tried to apply non-parametric statistics and machine learning techniques, such as ν-SVR and polynomial kernel, to networking research. Not only theoretical approach, but also practice approach was used in this research for the improved prediction results.

7.4 Future Directions

There are other aspects of the research area beyond issues that are important in the network throughput prediction but were not covered in the SVR-based predictor. First, many experiments and analysis should be explored to improve predictability. Next, further research is needed for the temporal prediction because the prediction problem deals with a regression problem. Finally, the SVR-base predictor should be designed and implemented as a module for the meta-scheduler, an Internet application for path selection, or a migration on cloud service. These are some of the research issues that were not able to be covered in this thesis but merit future investigation.

7.4.1 Issue of Improving Predictability

The SVR-based predictor relies on throughput measurement results gathered in the past to predict the network throughput. Thus, a prediction result is assumed to be similar to the past, and a lack of historical data would be hard to predict the throughput precisely. In order to overcome such problem, a sophisticated mathematical model should be considered with adequate parameters. Next, our experiments were executed on PlanetLab. It is a virtualized network testbed on the Internet, but it cannot represent all of Internet environments. To improve predictability of the SVR-based predictor, experiments and analysis should be explored on the other Internet environments, such as InTrigger [118], GENI [119], and so on. Although our predictor is more accurate and robust than the CDF-based predictor, we cannot determine that our predictor should be more precise than all of predictors. The prediction results of the SVR-based predictor should be compared to other predictors.

We focused on the problem of throughput prediction as a regression problem. From now on, the problem should deal with a temporal prediction to predict future throughput. Because it is more harder than the regression, an alternative measurement method based on the connection pair should be proposed to gather continuous time-series data, and a sophisticated model should be considered for the temporal prediction.

7.4.2 Issue of Design and Implements

In this thesis, we mainly presented the analytic model to build the SVR-based predictor and validity of prediction results by the simulation. The SVR-based predictor targets distributed, grid, and cloud computing.

For distributed systems, it can provide path state on multiple paths, and a user can select the best path that has the shortest transfer time. The SVR-based predictor should be designed and implemented as an Internet application. Next, we clearly showed the validation of prediction results through the meta-scheduler simulation. Since data-intensive grid tasks are rapidly increased, the SVR-based predictor should be implemented as a module on meta-scheduler. Thus, precise prediction results can reduce the processing time of given tasks. Moreover, imprecise prediction results should be also considered for the meta-scheduler. Particularly, re-scheduling mechanism should be considered when the imprecise prediction result is provided by the predictor. In cloud services, a migration is an ordinary method to move a virtual machine between different physical machines. In order to move the virtual machine, network resources have been considered. Thus, the prediction results can be used to reduce the migration time. Moreover, it can be also used in logging-replay [120] on distributed resources. We can determine the timing of logging-replay with the prediction results. For example, the logging-replay can be executed when network state is stable. Conversely, the logging-replay can be stopped by predicting the throughput when network state is unstable.

7.5 Closing Statement

The Internet has been evolved as a platform to support emerging applications for personal, business, enterprise, eScience, and so on. The throughput prediction can help the evolution and one of reasonable solutions for the efficient use of the Internet. In this thesis, we focus on measurements, analysis, and modeling for the precise throughput prediction on the Internet. The SVR-based predictor is accurate, robust, and suitable in comparison with the CDF-based predictor. It can support path selection on multi paths, the migration on cloud services, and grid schedulers as a criterion for a reduction of processing time. Finally, the research of throughput prediction will be emphasized for the efficient use of the Internet.

Bibliography

[1] Globus online, https://www.globusonline.org/.

[2] GridFTP, http://www.globus.org/toolkit/docs/latest-stable/gridftp/.

[3] Dropbox, https://www.dropbox.com/.

[4] Amazon EC2, http://aws.amazon.com/ec2/.

[5] Ian Foster and Carl Kesselman. *The Grid 2: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003. ISBN 1558609334.

[6] Luis Tomás, Agustín C. Caminero, Carmen Carrión, and Blanca Caminero. Network-aware meta-scheduling in advance with autonomous self-tuning system. *Future Gener. Comput. Syst.*, 27(5):486–497, May 2011. ISSN 0167-739X.

[7] Valliyammai Chinnaiah and Thamarai Selvi Somasundaram. A grid resource brokering strategy based on resource and network performance in grid. *Future Gener. Comput. Syst.*, 28(3):491–499, March 2012. ISSN 0167-739X.

[8] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield. Xen and the art of virtualization. In *Proceedings of the nineteenth ACM symposium on Operating systems principles*, SOSP '03, pages 164–177, New York, NY, USA, 2003. ACM. ISBN 1-58113-757-5.

[9] Jim Smith and Ravi Nair. *Virtual Machines: Versatile Platforms for Systems and Processes (The Morgan Kaufmann Series in Computer Architecture and Design)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005. ISBN 1558609105.

[10] Xen, http://www.xen.org/.

[11] Linux-VServer, http://linux-vserver.org/.

[12] VMWare, http://www.vmware.com/.

[13] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics).* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 0387310738.

[14] Benoit Claise, Ganesh Sadasivan, Vamsi Valluri, and Martin Djernaes. Cisco Systems NetFlow Services Export Version 9. RFC 3954, Internet Engineering Task Force (IETF), Oct. 2004.

[15] Jon Postel. Internet Control Message Protocol. RFC 792, Internet Engineering Task Force (IETF), Sep. 1981.

[16] Rich Wolski, Neil T. Spring, and Jim Hayes. The network weather service: a distributed resource performance forecasting service for metacomputing. *Future Gener. Comput. Syst.*, 15(5-6):757–768, October 1999. ISSN 0167-739X.

[17] Martin Swany and Rich Wolski. Multivariate resource performance forecasting in the network weather service. In *Supercomputing '02: Proceedings of the 2002 ACM/IEEE conference on Supercomputing*, pages 1–10, Los Alamitos, CA, USA, 2002. IEEE Computer Society Press.

[18] Mariyam Mirza, Joel Sommers, Paul Barford, and Xiaojin Zhu. A machine learning approach to TCP throughput prediction. *IEEE/ACM Trans. Netw.*, 18:1026–1039, August 2010. ISSN 1063-6692.

[19] PlanetLab, https://www.planet-lab.org/.

[20] Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and kyoji Umemura. Analysis of anomalies on a virtualized network testbed. In *CIT '10: Proceedings of the 10th IEEE International Conference on Computer and Information Technology*, pages 297–304, Washington, DC, USA, 2010. IEEE Computer Society. ISBN 978-0-7695-4108-2.

[21] Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura. Traffic anomaly analysis and characteristics on a virtualized network testbed. *IEICE Transactions on Information and Systems*, E94-D(12):2353–2361, 2011. ISSN 1745-1361.

[22] Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura. Internet traffic characteristics of virtual machine on Amazon EC2. *IPSJ SIG Notes*, 2011 (19):1–7, 2011.

[23] Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura. Estimating traffic anomalies for throughput prediction on network virtualization. In *Proceedings of the 2011 IEEE/IPSJ International Symposium on Applications and the Internet*, SAINT '11, pages 267–273, Washington, DC, USA, 2011. IEEE Computer Society. ISBN 978-0-7695-4423-6.

[24] Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura. A statistical approach for selecting throughput prediction parameters on the internet. In *Ubiquitous Information Technologies and Applications (CUTE), 2011 Proceedings of the 6th International Conference*, pages 37–40. Korea Information Processing Society (KIPS), December 2011.

[25] Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura. Predicting network throughput for grid applications on network virtualization areas. In *Proceedings of the first international workshop on Network-aware data management*, NDM '11, pages 11–20, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-1132-8.

[26] Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura. Analytical modeling of network throughput prediction on the internet. *IEICE Transactions on Information and Systems*, E95-D(12):2870–2878, Dec. 2012.

[27] Andy Bavier, Mic Bowman, Brent Chun, David Culler, Scott Karlin, Steve Muir, Larry Peterson, Timothy Roscoe, Tammo Spalink, and Mike Wawrzoniak. Operating system support for planetary-scale network services. In *NSDI'04: Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation*, pages 19–32, Berkeley, CA, USA, 2004. USENIX Association.

[28] Untae Kim, Chanho Choi, Hyeonsang Eom, and Heon Y. Yeom. A study of network performance isolation in the xen virtualization environment. In *Ubiquitous Information Technologies and Applications (CUTE), 2011 Proceedings of the 6th International Conference*, pages 262–264. Korea Information Processing Society (KIPS), December 2011.

[29] Ardalan Kangarlou, Sahan Gamage, Ramana Rao Kompella, and Dongyan Xu. vSnoop: Improving tcp throughput in virtualized environments via acknowledgement offload. In *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '10, pages 1–11, Washington, DC, USA, 2010. IEEE Computer Society. ISBN 978-1-4244-7559-9.

[30] Larry Peterson, Andy Bavier, Marc E. Fiuczynski, and Steve Muir. Experiences building planetlab. In *OSDI '06: Proceedings of the 7th symposium on Operating systems design and implementation*, pages 351–366, Berkeley, CA, USA, 2006. USENIX Association. ISBN 1-931971-47-1.

[31] Neil Spring, Larry Peterson, Andy Bavier, and Vivek Pai. Using planetlab for network research: myths, realities, and best practices. *SIGOPS Oper. Syst. Rev.*, 40(1):17–24, 2006. ISSN 0163-5980.

[32] Emulab, http://www.netbed.org/.

[33] StarBED, http://www.starbed.org/.

[34] Rahul Trivedi, Abhishek Chandra, and Jon Weissman. Heterogeneity-aware workload distribution in donation-based grids. *International Journal of High Performance Computing Applications*, 20:455–466, November 2006. ISSN 1094-3420.

[35] Joel Sommers and Paul Barford. An active measurement system for shared environments. In *IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 303–314, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-908-1.

[36] Dong Lu, Yi Qiao, Peter A. Dinda, and Fabian E. Bustamante. Characterizing and predicting tcp throughput on the wide area network. In *ICDCS '05: Proceedings of the 25th IEEE International Conference on Distributed Computing Systems*, pages 414–424, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2331-5.

[37] Thomas Anderson, Larry Peterson, Scott Shenker, and Jonathan Turner. Overcoming the internet impasse through virtualization. *Computer*, 38(4):34–41, 2005. ISSN 0018-9162.

[38] N.M. Mosharaf Kabir Chowdhury and Raouf Boutaba. A survey of network virtualization. *Comput. Netw.*, 54(5):862–876, 2010. ISSN 1389-1286.

[39] Gregor Schaffrath, Christoph Werle, Panagiotis Papadimitriou, Anja Feldmann, Roland Bless, Adam Greenhalgh, Andreas Wundsam, Mario Kind, Olaf Maennel, and Laurent Mathy. Network virtualization architecture: proposal and initial prototype. In *Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures*, VISA '09, pages 63–72, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-595-6.

[40] Jorge Carapinha and Javier Jiménez. Network virtualization: a view from the bottom. In *Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures*, VISA '09, pages 73–80, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-595-6.

[41] Thomas J. Hacker, Brian D. Athey, and Brian Noble. The end-to-end performance effects of parallel tcp sockets on a lossy wide-area network. In *IPDPS '02: Proceedings of the 16th International Parallel and Distributed Processing Symposium*, page 314, Washington, DC, USA, 2002. IEEE Computer Society. ISBN 0-7695-1573-8.

[42] Eitan Altman, Dhiman Barman, Bruno Tuffin, and Milan Vojnovic. Parallel tcp sockets: Simple model, throughput and validation. In *Proceedings of the 25th conference on Information communications*, INFOCOM'06, pages 1–12, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 3-937201-01-7.

[43] Slicestat, http://codeen.cs.princeton.edu/slicestat/.

[44] Vern Paxson and Mark Allman. Computing tcp's retransmission timer. RFC 2988, Internet Engineering Task Force (IETF), 2000.

[45] Massimiliano Rak, Rocco Aversa, Beniamino Martino, and Antonio Sgueglia. Web services resilience evaluation using lds load dependent server models. *Journal of Communications*, 5(1):39–49, 2010. ISSN 1796-2021.

[46] James F. Kurose and Keith W. Ross. *Computer Networking: A Top-Down Approach*. Addison-Wesley Publishing Company, USA, 4th edition, 2008. ISBN 0321497708, 978-0321497703.

[47] Amazon Web Services(AWS), http://aws.amazon.com/.

[48] Ru Iosup, Simon Ostermann, Nezih Yigitbasi, Radu Prodan, Thomas Fahringer, and Dick Epema. An early performance analysis of cloud computing services for scientific computing. *TU Delft, Tech. Rep., Dec 2008, [Online] Available*, 2008.

[49] Jiang Dejun, Guillaume Pierre, and Chi-Hung Chi. Ec2 performance analysis for resource provisioning of service-oriented applications. In *Proceedings of the 2009 international conference on Service-oriented computing*, ICSOC/ServiceWave'09, pages 197–207, Berlin, Heidelberg, 2009. Springer-Verlag. ISBN 3-642-16131-6, 978-3-642-16131-5.

[50] Guohui Wang and T. S. Eugene Ng. The impact of virtualization on network performance of amazon ec2 data center. In *INFOCOM'10: Proceedings of the 29th conference on Information communications*, pages 1163–1171, Piscataway, NJ, USA, 2010. IEEE Press. ISBN 978-1-4244-5836-3.

[51] sysstat, http://sebastien.godard.pagesperso-orange.fr/.

[52] Amazon CloudWatch, http://aws.amazon.com/cloudwatch/.

[53] Srikanth Kandula, Sudipta Sengupta, Albert Greenberg, Parveen Patel, and Ronnie Chaiken. The nature of data center traffic: measurements & analysis. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, IMC '09, pages 202–208, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-771-4.

[54] Mohammad Alizadeh, Albert Greenberg, David A. Maltz, Jitendra Padhye, Parveen Patel, Balaji Prabhakar, Sudipta Sengupta, and Murari Sridharan. Data center tcp (DCTCP). In *Proceedings of the ACM SIGCOMM 2010 conference on SIGCOMM*, SIGCOMM '10, pages 63–74, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0201-2.

[55] CoMon, http://comon.cs.princeton.edu/.

[56] KyoungSoo Park and Vivek S. Pai. Comon: a mostly-scalable monitoring system for planetlab. *SIGOPS Oper. Syst. Rev.*, 40(1):65–74, 2006. ISSN 0163-5980.

[57] Abhishek Chandra, Rohini Prinja, Sourabh Jain, and ZhiLi Zhang. Co-designing the failure analysis and monitoring of large-scale systems. *SIGMETRICS Perform. Eval. Rev.*, 36(2):10–15, 2008. ISSN 0163-5999.

[58] Kensuke Fukuda, Toshio Hirotsu, Osamu Akashi, and Toshiharu Sugawara. A pca analysis of daily unwanted traffic. In *Proceedings of the 2010 24th IEEE International Conference on Advanced Information Networking and Applications*, AINA '10, pages 377–384, Washington, DC, USA, 2010. IEEE Computer Society. ISBN 978-0-7695-4018-4.

[59] Haakon Ringberg, Augustin Soule, Jennifer Rexford, and Christophe Diot. Sensitivity of pca for traffic anomaly detection. In *SIGMETRICS '07: Proceedings of the 2007 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 109–120, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-639-4.

[60] I. T. Jolliffe. Discarding variables in a principal component analysis. i: Artificial data. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 21(2): 160–173, 1972. ISSN 00359254.

[61] Raymond B. Cattell. Factor analysis: An introduction to essentials ii. the role of factor analysis in research. 21:405–435, 1965. ISSN 0006-341X.

[62] Tatsuya Mori, Ryoichi Kawahara, Shozo Naito, and Shigeki Goto. On the characteristics of internet traffic variability: Spikes and elephants. *IEICE Transactions on Information and Systems*, 87-D(12):2644–2653, 2004. ISSN 0916-8532.

[63] K. Papagiannaki, N. Taft, S. Bhattacharyya, P. Thiran, K. Salamatian, and C. Diot. A pragmatic definition of elephants in internet backbone traffic. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurement*, IMW '02, pages 175–176, New York, NY, USA, 2002. ACM. ISBN 1-58113-603-X.

[64] Yin Zhang, Lee Breslau, Vern Paxson, and Scott Shenker. On the characteristics and origins of internet flow rates. In *Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '02, pages 309–322, New York, NY, USA, 2002. ACM. ISBN 1-58113-570-X.

[65] Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura. Evaluation of the throughput measurement method using connection pairs. In *WIT '09: In Proceedings of The Tenth Workshop on Internet Technology*. Japan Society for Software Science and Technology (JSSST), 2009.

[66] Aimin Sang and San qi Li. A predictability analysis of network traffic. *Computer Networks*, 39(4):329–345, 2002.

[67] Paulo Cortez, Miguel Rio, Pedro Sousa, and Miguel Rocha. Topology aware internet traffic forecasting using neural networks. In *Proceedings of the 17th international conference on Artificial neural networks*, ICANN'07, pages 445–454, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 3-540-74693-5, 978-3-540-74693-5.

[68] Hong Zhao. Multiscale analysis and prediction of network traffic. In *28th IEEE International Performance Computing and Communications Conference (IPCCC)*, pages 388–393, Washington, DC, USA, Dec. 2009. IEEE Computer Society. ISBN 9781424457373.

[69] Lin Xiang, Xiaohu Ge, Chuang Liu, Lei Shu, and Cheng-Xiang Wang. A new hybrid network traffic prediction method. In *Proceedings of the Global Communications Conference*, pages 1–5, Washington, DC, USA, 2010. IEEE Computer Society. ISBN 978-1-4244-5638-3.

[70] Mukul Goyal, Roch Guérin, and Raju Rajan. Predicting tcp throughput from non-invasive network sampling. In *Proceedings. IEEE INFOCOM 2002*, pages 180–189, Washington, DC, USA, Jun. 2002. IEEE Computer Society.

[71] Zalal Uddin Mohammad Abusina, Salahuddin Muhammad Salim Zabir, Ahmed Ashir, Debasish Chakraborty, Takuo Suganuma, and Norio Shiratori. An engineering approach to dynamic prediction of network performance from application logs. *Int. J. Netw. Manag.*, 15(3):151–162, 2005. ISSN 1099-1190.

[72] Qi He, Constantine Dovrolis, and Mostafa Ammar. On the predictability of large transfer tcp throughput. In *SIGCOMM '05: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 145–156, New York, NY, USA, 2005. ACM. ISBN 1-59593-009-4.

[73] Leszek Borzemski, Marta Kliber, and Ziemowit Nowak. Using data mining algorithms in web performance prediction. *Cybern. Syst.*, 40(2):176–187, 2009. ISSN 0196-9722.

[74] Sudharshan Vazhkudai, Jennifer M. Schopf, and Ian T. Foster. Predicting the performance of wide area data transfers. In *IPDPS '02: Proceedings of the 16th International Parallel and Distributed Processing Symposium*, page 270, Washington, DC, USA, 2002. IEEE Computer Society. ISBN 0-7695-1573-8.

[75] Dengpan Yin, Esma Yildirim, and Tevfik Kosar. A data throughput prediction and optimization service for widely distributed many-task computing. In *Proceedings of the 2nd Workshop on Many-Task Computing on Grids and Supercomputers*, MTAGS '09, pages 4:1–4:10, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-714-1.

[76] Jae-Hyun Hwang and Chuck Yoo. Formula-based tcp throughput prediction with available bandwidth. *Comm. Letters.*, 14(4):363–365, April 2010. ISSN 1089-7798.

[77] Muhammad Murtaza Yousaf, Michael Welzl, and Malik Muhammad Junaid. Fog in the network weather service: a case for novel approaches. In *Proceedings of the first international conference on Networks for grid applications*, GridNets '07, pages 231–236, ICST, Brussels, Belgium, Belgium, 2007. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering). ISBN 978-963-9799-02-8.

[78] Sudharshan Vazhkudai and Jennifer M. Schopf. Predicting sporadic grid data transfers. In *HPDC '02: Proceedings of the 11th IEEE International Symposium on High Performance Distributed Computing*, page 188, Washington, DC, USA, 2002. IEEE Computer Society. ISBN 0-7695-1686-6.

[79] Paola Bermolen and Dario Rossi. Support vector regression for link load prediction. *Comput. Netw.*, 53:191–201, February 2009. ISSN 1389-1286.

[80] Robert Beverly, Karen Sollins, and Arthur Berger. SVM learning of IP address structure for latency prediction. In *Proceedings of the 2006 SIGCOMM workshop on Mining network data*, MineNet '06, pages 299–304, New York, NY, USA, 2006. ACM. ISBN 1-59593-569-X.

[81] Huifang Feng, Yantai Shu, and Maode Ma. WLAN traffic prediction using support vector machine. *IEICE Transactions on Communications*, 92(9):2915–2921, 2009-09-01. ISSN 09168516.

[82] David Andersen, Hari Balakrishnan, Frans Kaashoek, and Robert Morris. Resilient overlay networks. *SIGOPS Oper. Syst. Rev.*, 35:131–145, October 2001. ISSN 0163-5980.

[83] Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995. ISBN 0-387-94559-8.

[84] Bernhard Schölkopf, Alex J. Smola, Robert C. Williamson, and Peter L. Bartlett. New support vector algorithms. *Neural Comput.*, 12(5):1207–1245, May 2000. ISSN 0899-7667.

[85] A. Aizerman, E. M. Braverman, and L. I. Rozoner. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

[86] e1071:Misc Functions of the Department of Statistics http://cran.r-project.org/web/packages/e1071/.

[87] The R Project for Statistical Computing http://www.r-project.org/.

[88] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27:1–27:27, May 2011. ISSN 2157-6904.

[89] Trevor Hastie and Robert Tibshirani. Generalized additive models. *Statistical Science*, 1(3):297–318, 1986.

[90] Simon N. Wood. *Generalized Additive Models: An Introduction with R*. Chapman & Hall/CRC, 2006. ISBN 1-584-88474-6.

[91] Jerome H. Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, 19(1):1–67, 1991. ISSN 00905364.

[92] gam: Generalized Additive Models http://cran.r-project.org/web/packages/gam/index.html.

[93] Hiroyuki Ohsaki, Kazunori Nozaki, Kenichi Baba, Eisaku Sakane, Naohisa Sakamoto, Koji Koyamada, and Shinji Shimojo. Peta-flow computing: Vision and challenges. In *SAINT*, pages 256–259, Washington, DC, USA, 2011. IEEE Computer Society. ISBN 978-1-4577-0531-1.

[94] Large Hadron Collider (LHC), http://public.web.cern.ch/public/en/LHC/.

[95] Large Synoptic Survey Telescope (LSST), http://www.lsst.org/lsst/about.

[96] D. Bader. *Petascale Computing: Algorithms and Applications*. Chapman and Hall/CRC, London, 1 edition, December 2008. ISBN 1584889098.

[97] Uniform Interface to Computing Resources (UNICORE), http://www.unicore.eu/.

[98] Globus, http://www.globus.org/.

[99] Atsuko Takefusa, Michiaki Hayashi, Naohide Nagatsu, Hidemoto Nakada, Tomohiro Kudoh, Takahiro Miyamoto, Tomohiro Otani, Hideaki Tanaka, Masatoshi Suzuki, Yasunori Sameshima, Wataru Imajuku, Masahiko Jinno, Yoshihiro Takigawa, Shuichi Okamoto, Yoshio Tanaka, and Satoshi Sekiguchi. G-lambda: coordination of a grid scheduler and lambda path service over gmpls. *Future Gener. Comput. Syst.*, 22:868–875, October 2006. ISSN 0167-739X.

[100] Atsuko Takefusa, Hidemoto Nakada, Tomohiro Kudoh, Yoshio Tanaka, and Satoshi Sekiguchi. GridARS: an advance reservation-based grid co-allocation framework for distributed computing and network resources. In *Proceedings of the 13th international conference on Job scheduling strategies for parallel processing*, JSSPP'07, pages 152–168, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 3-540-78698-8, 978-3-540-78698-6.

[101] Ming Wu and XianHe Sun. Grid harvest service: a performance system of grid computing. *J. Parallel Distrib. Comput.*, 66:1322–1337, October 2006. ISSN 0743-7315.

[102] Srikumar Venugopal, Rajkumar Buyya, and Lyle Winton. A grid service broker for scheduling e-science applications on global data grids: Research articles. *Concurr. Comput. : Pract. Exper.*, 18:685–699, May 2006. ISSN 1532-0626.

[103] Agustín Caminero, Omer Rana, Blanca Caminero, and Carmen Carrión. Network-aware heuristics for inter-domain meta-scheduling in grids. *J. Comput. Syst. Sci.*, 77(2):262–281, March 2011. ISSN 0022-0000.

[104] Arturo Crespo and Hector Garcia-Molina. Routing indices for peer-to-peer systems. In *Proceedings of the 22 nd International Conference on Distributed Computing Systems (ICDCS'02)*, ICDCS '02, pages 23–32, Washington, DC, USA, 2002. IEEE Computer Society. ISBN 0-7695-1585-1.

[105] DataGrid, http://eu-datagrid.web.cern.ch/eu-datagrid/.

[106] Richard McClatchey, Ashiq Anjum, Heinz Stockinger, Arshad Ali, Ian Willers, and Michael Thomas. Data intensive and network aware (DIANA) grid scheduling. *Journal of Grid Computing*, 5(1):43–64, 2007. ISSN 1572-9184.

[107] Ching-Hsien Hsu, Chia-Wei Chu, and Chih-Hsun Chou. Bandwidth sensitive co-allocation scheme for parallel downloading in data grid. In *Proceedings of*

the International Symposium on Parallel and Distributed Processing with Applications, ISPA '09, pages 34–39, Washington, DC, USA, 2009. IEEE Computer Society. ISBN 978-0-7695-3747-4.

[108] Konstantinos Christodoulopoulos, Nikolaos Doulamis, and Emmanouel Varvarigos. Joint communication and computation task scheduling in grids. In *Proceedings of the 2008 Eighth IEEE International Symposium on Cluster Computing and the Grid*, pages 17–24, Washington, DC, USA, 2008. IEEE Computer Society. ISBN 978-0-7695-3156-4.

[109] The Network Simulator (NS2), http://isi.edu/nsnam/ns/.

[110] Yukio Tsukishima, Michiaki Hayashi, Tomohiro Kudoh, Akira Hirano, Takahiro Miyamoto, Atsuko Takefusa, Atsushi Taniguchi, Shuichi Okamoto, Hidemoto Nakada, Yasunori Sameshima, Hideaki Tanaka, Fumihiro Okazaki, and Masahiko Jinno. Grid network service-web services interface version 2 achieving scalable reservation of network resources across multiple network domains via management plane. *IEICE Transactions on Communications*, E93.B(10): 2696–2705, 2010.

[111] JGN http://www.jgn.nict.go.jp/english/index.html.

[112] Eric Mannie. Generalized multi-protocol label switching (GMPLS) architecture. RFC 3945, Internet Engineering Task Force (IETF), 2004.

[113] P. Kokkinos and E. A. Varvarigos. Scheduling efficiency of resource information aggregation in grid networks. *Future Gener. Comput. Syst.*, 28(1):9–23, January 2012. ISSN 0167-739X.

[114] Nikos Doulamis, Panagiotis Kokkinos, and Emmanouel Varvarigos. Spectral clustering scheduling techniques for tasks with strict qos requirements. In *Proceedings of the 14th international Euro-Par conference on Parallel Processing*, Euro-Par '08, pages 478–488, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-85450-0.

[115] Feng Liang, Shilong Ma, Andre Luckow, and Bettina Schnor. Earliest start time estimation for advance reservation-based resource brokering within computational grids. In *Proceedings of the International Symposium on Parallel and Distributed Processing with Applications*, ISPA '10, pages 8–15, Washington, DC, USA, 2010. IEEE Computer Society. ISBN 978-0-7695-4190-7.

[116] SETI@home, http://setiathome.berkeley.edu/.

[117] Tracy D. Braun, Howard Jay Siegel, Noah Beck, Lasislau L. Bölöni, Muthucumara Maheswaran, Albert I. Reuther, James P. Robertson, Mitchell D. Theys, Bin Yao, Debra Hensgen, and Richard F. Freund. A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. *J. Parallel Distrib. Comput.*, 61(6):810–837, June 2001. ISSN 0743-7315.

[118] InTrigger https://www.intrigger.jp/wiki/index.php/InTrigger.

[119] Global Environment for Network Innovations (GENI), http://www.geni.net/.

[120] Kei Ohmura, Yoshiaki Tamura, Toru Yuguchi, and Satoshi Moriai. Rapid vm synchronization with I/O emulation logging-replay (in Japanese). *IPSJ SIG Notes*, 2011(16):1–8, 2011.

Acknowledgements

Many friends and colleagues have contributed to my research work, and I thank them all here.

First, I would like to thank my first supervisor, Professor Kyoji Umemura for his guidance, support, patience, and encouragement, which enabled me to overcome many obstacles and finish my thesis. I am deeply indebted to my second supervisor at Professor Toshio Hirotsu at the Faculty of Computer and Information Sciences, Hosei University for his time, support and professional assistance. He introduced me to the deep sea of research and led to research areas at undergraduate student. I highly acknowledge the marvelous support, encouragement and counseling from Associate Professor Hirotake Abe at the Department of Computer Science, University of Tsukuba who is my third supervisor. He always supported me morally and technically when I used to be in trouble, and helped like an elder brother.

Second, I should thank Professor Masaki Aono who is the chairman of dissertation committee and Professor Hideyuki Uehara who is a member of dissertation committee for their helpful comments and advices.

Third, thanks to Assistant Professor Masayuki Okabe and members of Umemura Laboratory for various kinds of support and help from them, I have finished my thesis. I wish to express my sincere appreciation to Ms. Masako Hata who is secretary of Umemura Laboratory. I also very appreciate JSPS KAKENHI, the Global COE program for many research supports, and the Ministry of Education, Culture, Sports, Science, and Technology for financial support through the award of Monbukagakusho Scholarship.

Finally, I will not be doing justice if I cannot acknowledge the tireless services, care and support of my wife Sunock Chang throughout my research work. Her patience, countless sacrifices and understanding remained a prime catalyst for the timely completion of this work. I should also be very grateful to Ms. Yukiko Shimizu who always remained supportive for me, and helped like a mother. I am indebted to my parents and parents-in-law. All of them provided me with constant understanding, encouragement, and support.

January 2013
Chunghan Lee

List of publications

- Journals
  - Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura, "Analytical Modeling of Network Throughput Prediction on the Internet", IEICE Trans. (D), Vol.E95-D, No.12, pp. 2870 - 2878, Dec. 2012.
  - Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura, "Traffic Anomaly Analysis and Characteristics on a Virtualized Network Testbed", IEICE Trans. (D), Vol.E94-D, No.12, pp. 2353 - 2361, Dec. 2011.
- International conferences with referred
  - Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura, "Performance Implications of Task Scheduling by Predicting Network Throughput on the Internet", The 11th IEEE International Symposium on Parallel and Distributed Processing with Applications (ISPA-13), (submitted 2013/03/01).
  - Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura, "A Statistical Approach for Selecting Throughput Prediction Parameters on the Internet", The 6th Edition of the International Conference on Ubiquitous Information Technologies & Applications (CUTE 2011), pp. 37 - 40, Seoul, Korea, Dec. 2011.
  - Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura, "Predicting Network Throughput for Grid Applications on Network Virtualization Areas", Proc. IEEE/ACM SC'11 The first International Workshop on Network-Aware Data Management (NDM 2011), pp. 11 - 20, Seattle, USA, Nov. 2011.
  - Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura, "Estimating Traffic Anomalies for Throughput Prediction on Network Virtualization", Proc. IEEE/IPSJ SAINT'11 The Second Workshop on High Speed Network and Computing Environments (HSNCE 2011), pp. 267 - 273, Munich, Germany, Jul. 2011.
  - Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura, "Analysis of Anomalies on a Virtualized Network Testbed", The 10th IEEE International Conference on Computer and Information Technology (CIT'10), pp. 297 - 304, Bradford, UK, Jun. 2010.

- International conferences
  - Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura, "A Proposal for Improving Network Throughput Predictability over the Internet", The Pacific Rim Application and Grid Middleware Assembly (PRAGMA) 21 workshop, Sapporo, Japan, Oct. 2011. [Poster presentation]
  - Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura, "Characteristics of Internet Traffic Anomalies over a Virtualized Network Testbed", The 7th Korea-Japan e-Science Symposium, Hongcheon, Korea, Jul. 2010.
- Domestics conferences with referred
  - Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura, "Recent Empirical Evaluation of Aggregate Throughput using Parallel Transfers on the Internet", Computer & System Symposium 2012 (ComSys2012), pp. 53 - 58, Tokyo, Japan, Dec. 2012.
  - Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura, "A PCA Analysis for Traffic Anomaly Estimation", Internet Conference 2010 (IC2010), pp. 93 - 98, Tokyo, Japan, Oct. 2010.
  - Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura, "Evaluation of the Throughput Measurement Method using Connection Pairs", The 10th Workshop on Internet Technology (WIT2009), Furano, Japan, Jun. 2009.
- Domestics conference
  - Chunghan Lee, Hirotake Abe, Toshio Hirotsu, and Kyoji Umemura, "Internet Traffic Characteristics of Virtual Machine on Amazon EC2", IPSJ SIG Notes (SIGOS 117), pp. 1 - 7, Okinawa, Japan, Apr. 2011.
- Research Award
  - Best Paper Honorable Mention in NDM2011, the 1st workshop on Network-aware Data Management (held in conjunction with IEEE/ACM SC11)