# Studies on Network Traffic Engineering Exploiting Dynamic IP-Flow Characteristics

**(動的な IP フローの特性を利用したトラヒックエンジニアリング技術に関する研究)**

Yoshinori KITATSUJI

# Preface

Due to advances of the Internet, various distributed applications are commonly used and thus, traffic type traversing the Internet diversifies according with increase of the application variation. Network traffic engineering (TE) which distributes traffic over multiple routes (paths), efficiently utilizes the network resource and relieves the contention of the network resource. Such effectiveness rises when TE applied on a backbone network admitting huge amounts of traffic. The conventional methods for TE generally take the utilization of individual paths into account for traffic distribution. However, there is a nature that even constant bit-rate traffic suffers from high delay variation as much as the highly burst traffic does when a path involves both types of traffic, if its utilization is high. This dissertation discusses TE which distinguishes flows from traffic to have a certain traffic characteristics and reduces delay variation of the flows as a whole by exploiting the characteristics when the flows are distributed over multiple paths.

For this study, the future backbone network is considered to have two layer structure, a core network and edge routers surrounding it, for the sake of dealing with huge amounts of traffic, e.g., thousands Gbit/s or more in total. It is assumed that multiple bandwidth-guaranteed paths are established between each pair of the edge routers and routed over the core network. And the edge routers handles assigning flows to one of the paths to distribute traffic over the core network. In this assumption, the edge routers are considered to have two functions, measuring the traffic characteristics of flows and assigning flows to one of paths with their characteristics. This dissertation studies the individual functions along with the following steps. And thus, it is indicated that the TE exploiting traffic characteristics effectively improves the quality of services (QoS).

First, the availability of the real-time IP-flow monitoring on the high-speed link, e.g., 10 Gbit/s, is studied through developing a traffic monitoring tool dealing with huge amounts of flows. Second, the effectiveness of TE focusing on traffic flows is studied through two analyses using the real traffic with the developed IP-flow monitoring tool. One analysis is for TCP flows of GRID applications. This analysis indicates that when multiple GRID applications simultaneously run in a network, effective bandwidth assignment for a set of flows from applications differs depending on their traffic patterns. E.g. application traffic including highly bursty flows should have an exclusive path since it degrades the other application performance if aggregated to a single path, while application traffic with small changes of traffic rate can be aggregated into a single path with other similar applications without large degradation of application-level performance. The other analysis is for UDP flows with VoIP traffic. This analysis finds that VoIP traffic mainly involves distinctive flows, that is, fix packet length and almost fix inter-packet gap. The analysis leads to developing a lightweight VoIP flow monitoring method exploiting the VoIP flow features. Third, a flow assignment method for reducing the queuing delay in the case of two distinctive flow sets assigned to two different bandwidth paths is developed. The contribution of this method is that it finds the approximate flow assignment within a short period so as to be used in the real environment.

Findings clarified in these studies are 1) feasibility of developing scalable flow-based traffic measurement method, 2) effectiveness of introducing the traffic characteristics on individual flows acquired by traffic measurement into the TE, and 3) effectiveness of a developed flow assignment method exploiting the traffic characteristics on flows. These results greatly contribute to effective improvement of QoS in TE.

Finally, I hope that this dissertation will be helpful for further studies in this field.

March 2007

Yoshinori KITATSUJI

# Acknowledgements

First and foremost, I thank my family, especially my wife, Hidemi, who challenged me to "do the best you can, then let it go." Better advice has never been uttered.

I wish to express my sincere appreciation to Professor Masato Tsuru of Kyushu Institute of Technology. His constant encouragement, guidance through this research, valuable discussions and advice have greatly helped in accomplishing the research. I also thank him for his careful reading of all papers on these studies.

I would also like to express my gratitude to Professor Katsuyuki Yamazaki of Nagaoka University of Technology and Professor Yuji Oie of Kyushu Institute of Technology for their valuable comments, time, and help in completing these studies. Their steady supports have greatly helped my study.

I wish to thank Professor Tetsuya Takine of Osaka University, Assistant Professor of Hiroshi Koide of Kyushu Institute of Technology, and Mr. Satoshi Katsuno in KDDI R & D Laboratories, Inc. for their valuable discussions and comments.

I would like to express my gratitude to Mr. Kazunori Konishi in KDDI R & D Laboratories, Inc., Assistant Professor Akira Kato of University of Tokyo, Dr. Katsushi Kobayashi in National Institute of Advanced Industrial Science and Technology, and Mr. Yasuichi Kitamura in NICT concerning the work in Chapter 2 for their precise works and comments. Without their assistance and advice on network designing and operations, I would not be able to maintain Tokyo exchange point in APAN, and to construct knowledge of what IP network is and understanding how it works.

I wish to thank Mr. Junji Hirooka and Mr. Hiroaki Matsuo at Information and Communications Technology Section in Human Media Creation Center for their continuous devoted encouragements. Without them, I would not concentrate to drive myself to study in Kitakyushu where almost topics in this dissertation have been studied.

I extend thanks to Dr. Kazmumi Kumazoe, Dr. Hiroyuki Koga, Dr. Nobuo Ryoki and all other members of NICT Kyushu Research Center for their kindly supports and invaluable discussions.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Background and Scope of these Studies

## 1.1 Introduction

Due to advances of the Internet, various distributed applications are generally used and thus the characteristics of traffic traversing the Internet now diversifies according with increase of the application variation. So far, the Internet Service Providers (ISPs) had been facing increasing the capacity of admitted traffic, while in addition they simultaneously challenge to keep the traffic of the multiple services, such as, conventional text-based file transmission, voice-over-IP (VoIP), IP Television and etc, high quality and stationary when these services are provided in their single network.

One of the network traffic engineering (TE) using multiple communication paths (simply called *paths* hereafter) to distribute incoming traffic over a network can increase capacity of the admitted traffic. Conventionally, the traffic distribution equally balances traffic over the multiple paths based on the link capacity, that is, making the link utilization equalized [AAA+02] [SdV01] [SGD03]. Then, the distribution usually exploits destination IP address of the traffic, resulting in that multiple service traffic with the distinct characteristics, such as burst and constant bit-rate (non-burst) traffic, are assigned to an identical path. Thus, there is an essential problem that the burst traffic heavily increases the delay variation when the link is highly utilized, leading to that the non-burst traffic accommodated in the same path also encounters the heavy delay variation [SB02] [TV00] [XG02]. TE exploiting the traffic characteristics of flows, e.g., constant bit-rate and burstiness, for improving QoS when the path is highly utilized is studied in this dissertation. Here, a flow is defined as a sequence of packets distinguished by tuples, that is, the single or the multiple fields in IP- and transport-layers headers. The combination of tuples is depends on how the flows are distinguished from the others, such as, a flow distinguishing source and destination port in transport-layer as described in Chapter 5, or a flow distinguishing applications executed over multiple hosts as described in Chapter 4

A backbone dealing with huge amounts of traffic is considered to have an two-layer structure, a core network to aggregate traffic traversing the backbone network, and edge routers where the traffic enters or leaves the backbone network. Exclusive routers having simple packet-forwarding mechanism are placed in the core network leading to exchanging huge amounts of traffic, while the edge router has higher functionality (more complex processing) than ones in the core network, since the traffic treated by the former router is lower than the latter. In addition, the individual pair of the edge routers establishes multiple bandwidth-guaranteed paths routed over the core network in order to properly distribute traffic forwarded through the paths over the core network. In this structure, the individual edge router is required to equip two functions: measuring of the traffic characteristics of each flow, and appropriately balancing the flows over the multiple paths by assigning flows to the paths with the identified traffic characteristics. For this traffic characteristic-aware TE, two-folds are studied:

- a method measuring the flows traversing a high-speed link, and
- a method properly balancing flows over multiple paths in terms of a queuing delay in a buffer of the first link of each path.

The scalable flow measurement method is studied through developing a real-time IP-flow monitoring tool. The efficiency of the flow measurement is indicated through the analysis of two types of applications with the developed tool: distributed computing applications using TCP flows, and VoIP traffic composed of UDP flows. For the TCP-flow analysis the relationship of network properties and application traffic is analyzed in terms of improving the application-level performance (completion time) of the distributed computing applications. The fact that the effective way to allocate a bandwidth-guaranteed path to the single or multiple applications depends on the traffic pattern of them is found when multiple types of the applications run over a single network. For the UDP-flow analysis a lightweight method exposing VoIP flows and performance metrics representing the QoS level of VoIP flows are developed.

For the study on the queuing delay-aware flow distribution, a flow balancing method for the simple situation that two types of flows with distinct characteristics are balanced over two different bandwidth paths through the following two steps. The first step is confirming that balancing the ideal flows grouped into two types over two paths can reduce queuing delay by some tens folds compared to the path utilization-based balancing. The second step is developing a method finding the approximately optimal flow assignment exploiting several sample data of the previous flow assignment and its queuing delay without the information on the detailed traffic characteristics. The method enables the flow assignment to reduce the queuing delay by multiple folds.

Through these studies, it is proved that the following three-folds are effective for preventing the severe quality-degradation when flows are admitted to the highly utilized backbone network: the scalable flow measurement, the TE focusing on exploiting the traffic characteristics observed by the developed flow measurement tool. and the generic flow-balancing method taking into account the traffic characteristics of flows.

## 1.2 Network Traffic Engineering

The Internet has been growing, as real-time traffic, such as, VoIP, IPTV, and so forth, has been increasingly conveyed besides the conventional file transmission in the Internet. The backbone network, e.g., the Internet service providers (ISPs), faces to emerge providing various types of traffic with the QoS required by their customers.

TE properly and dynamically allocates the network resources to such various types of traffic according with the network situation [AMA+99]. Although there exist variety of TE definition, the goals of TE are grouped into the followings:

1. improving the network fault tolerance
2. increasing the capacity of admitted traffic
3. improving the QoS of input traffic as a whole or a part
4. guaranteeing the QoS of admitted traffic.

1. avoids the communication inavailability, such as, discarded or loops by rerouting the traffic after the network problem happening. This is achieved by the effective protection and restoration of links or paths. The protection recoveries a fault with a pre-computed path while the restoration recoveries with a path created after detecting the fault. Conventionally, the link-level recovery was done for SONET and FDDH which continuously monitor link status and receiving frames. The path-level recovery was done by routing protocol and used to require long restoration time in the order of seconds to minutes. Multi-protocol label switching (MPLS) [AMA+99] is now one-solution to make the recovery more flexibly and faster [SE03]. There are various methods for finding the dedicated path for the protection and computing restoration path.

2. is achieved by optimizing link costs of link-state routing protocol, or balancing traffic over the multiple routes for each traffic path effectively [Kaw04] [SdV01]. Although the former method uses the single path for each traffic path, the capacity improves by 1.5 times more than that from the general cost setting, such as, in the in-proportional way to the link bandwidth [FT00]. In the latter cases, early traffic balancing is limited to be used with multiple equal-cost or relaxed equal-cost paths [SGD03], [Vil99]. These were for the sake of avoiding loop for packet forwarding using the routing table constructed by conventional defact-standard routing protocols, such as, RIP, OSPF, IS-IS. In the cases of use for label switch path (LSP) of MPLS, PVC of ATM, or VLAN of Ethernet, not only the shortest path but also the *k*-shortest paths algorithms are used to find an multiple paths. Topkis proposed

the enhanced algorithm of Djikstra so that OSPF discovers $k$ routes [Top] in the order of the low cost. The algorithm is given the maximum path cost $C$ and discovers the lowest cost route in all the routes without more than the path cost $C$. Chen proposed the $k$-largest bandwidth path algorithm exploiting the bandwidth of a link as a metric, where the best route has the largest metric in all the candidate routes [Che99].

3. avoids the degradation of traffic characteristics as a whole or a part of incoming traffic when whole traffic is accepted to be conveyed in the backbone network (without admission control). This type of TE also uses multiple paths as 2. does. Most difference from 2. is how balance traffic over the multiple paths. 3. generally equalizes the utilization of individual paths over which traffic is distributed, while 3. uses other criteria related to the delay variation related to the delay variation and packet loss with the measurement-based information on traffic characteristics [EJLW01] [GKL$^+$04]. In the 2. case, the network resource are optimally allocated under the variety of fairness definitions.

4. increases the admitted traffic which has variety of QoS requirements. The difference from 3. is the admission control is generally adopted to achieve the QoS requirement of the admitted traffic [KKL00]. 3. and 4. are called *measurement based routing* in the cases to compare with the conventional routing methods which do not use network dynamics or do use only the link utilization [GCL04a].

Studies in this dissertation are classified into 3. These studies first consider the structure of backbone network which deals with huge amounts of traffic and equips functions to measure traffic characteristics in it Second, the measurement method dealing with large amounts of flows is considered. Lastly, The flow assignment method balancing the delay variation over multiple paths is proposed.

## 1.3 Structure of These Studies

Figure 1.1 illustrates structure of these studies. The following are brief summary of each study and associated Chapter.

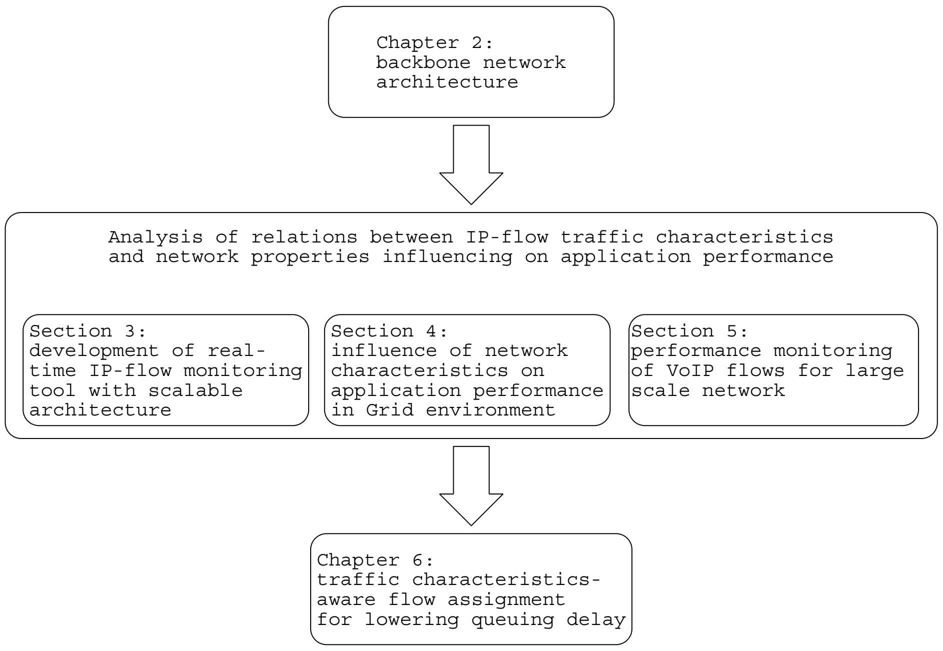

Chapter 2: backbone network architecture
Analysis of relations between IP-flow traffic characteristics and network properties influencing on application performance
Section 3: development of real-time IP-flow monitoring tool with scalable architecture
Section 4: influence of network characteristics on application performance in Grid environment
Section 5: performance monitoring of VoIP flows for large scale network
Chapter 6: traffic characteristics-aware flow assignment for lowering queuing delay

Figure 1.1: Scope and structure of these studies

- Chapter 2: *Backbone Network Architecture* describes Asia-Pacific Advanced Network (APAN) [LW00] and reviews the network configuration and operations of APAN Tokyo exchange point (XP), and policy based routing scheme with multiple routers adopted in Tokyo XP. The multiple router scheme is evaluated and shows to have the sufficient packet-forwarding performance compared to the requirement in Tokyo XP. The results discussed in this chapter are mainly taken from works [KKK+02]. Through these reviews and performance evaluation, the chapter clarifies the requirements for a backbone network providing the better QoS among traffics with the various traffic characteristics.

- Chapter 3: *Development of Real-time IP-flow Monitoring Tools with Scalable Architecture* describes the architecture for a scalable real-time flow measurement tool in order to allow network operators to flexibly define “the targeted flows” on-demand, to obtain various statistics on those flows, and to visualize them in a real-time manner. A traffic distribution device and multiple traffic capture devices processing packets in parallel are included in the architecture, in which the former device copies traffic and distributes it to the latter devices. The performance evaluation with a proto-type implementation on PC-UNIX in testbed experiments shows the proposed architecture has the scalability and the advantage of flexible fine-grained flow measurements. The results discussed in this chapter are mainly taken from works [KYTO04] [KKY+07].

- Chapter 4: *Influence of Network Characteristics on Application Performance in a Grid Environment* describes analysis of the relation between the characteristics of traffic generated by typical grid applications, and the effect of the round-trip time and bottleneck bandwidth of network on the application-level performance (i.e., completion time) of these applications. The analysis shows that the impact of network conditions on the performance of various applications and the impact of application traffic on network conditions differ considerably depending on the application. These results suggest that effective allocation of network resources must take into account the network-related properties of individual applications. The results discussed in this chapter are mainly taken from works [KYK+05].

- Chapter 5: *Performance Monitoring of VoIP Flows for Large Scale Network* describes a method for scalable performance monitoring of VoIP traffic flows distinguished by five tuples, that is, IP addresses, protocol, and port numbers. This method extracts the VoIP flows in a lightweight manner, and estimates the packet loss rate and the delay variation by exploiting the features of actual VoIP flows, that is, fixed packet length and fixed inter-packet gaps (IPGs) which are relatively common among most VoIP flows. The evaluation by simulation with trace data of actual traffic shows that monitoring only 2 % or less of the total VoIP flows can accurately estimate the delay variation of the total VoIP flows, while 20 % or more of the total VoIP flows are required in order to estimate the packet loss rate.

- Chapter 6: *Traffic Characteristic-aware Flow Assignment over Multiple Paths for lowering queuing delay* describes a novel traffic characteristic-aware flow assignment method to lower queuing delay in a fundamental case where two types of flows with distinct traffic characteristics (e.g., burstiness) are distributed into two paths. The method finds a nearly optimal flow assignment in terms of either minimizing the worst queuing delay among two paths or maximizing the sum of logarithm of the difference between a pre-defined queuing delay upper-limit and the actual queuing delay of individual paths weighted by traffic volume on each path. The evaluation suggests that considering the traffic characteristics significantly improves the delay performance in the flow distribution over multiple paths. The results discussed in this chapter are mainly taken from works [KKT+06] [KKT+07].

# Chapter 2

# Backbone Network Architecture

Asia-Pacific Advanced Network (APAN) [LW00] established in 1997 develops the number of interconnection, bandwidth of links composing APAN, reachability, and Service variation, and results in becoming recognized as an international testbed interconnecting Asia-Pacific and US regions. Although the conventional international testbeds often have a type of interconnecting a pair of countries with a point-to-point link, APAN interconnects the multiple point-to-point links at the exchange points (XP), such as, Tokyo and Seoul XPs, and results in providing a environment of the true international testbed interconnecting multiple countries in the Asia-Pacific region. The RENs and research institutions, which APAN interconnects with, are Singapore Advanced Research and Educational Network (SingAREN), China Education and Research Network (CERNET), Malaysia Supper Corridor (MSC), PHnet (Philippine), TransPAC [tra02], and StarTAP2 [Def01]. For Japanese RENs, APAN interconnects with Japan Gigabit Network 2 (JGN2), WIDE, National Institution of Information and Communications Technology (NICT), Science and Information Network (SINET), and some universities and research institutions.

In this chapter, first, the network configuration and operations of Tokyo XP is reviewed and, the policy-based routing (PBR) scheme adopted in Tokyo XP to combine multiple routers with insufficient capacity of maintaining multiple routing tables in Section 2.1.5 From these consideration, the requirements for a backbone network providing the better QoS among traffics with the various traffic characteristics than conventional link utility-based TE are clarified.

The reminder of this chapter is as follow. Section 2.1 summarizes Tokyo XP with describing its network configuration, routing, network operations, tools, PBR scheme. Section 2.2 evaluates the PBR scheme and shows the sufficient packet-forwarding performance compared to the requirements. In Section 2.3, the backbone network architecture is considerd for proving better QoS for a large amount of traffic (multiple-ten Gbit/s) with various traffic characteristics exchanged in a backbone network. Section 2.4 concludes this chapter with remarks.

## 2.1 Summary of APAN Tokyo XP

The network configuration, operation, tools to reduce the operational workload in APAN Tokyo XP in 2002, are presented in this section.

### 2.1.1 International Links and Acceptable Usage Policy

APAN interconnects the point-to-point testbed links at XPs and configures a multipoint testbed. Table 2.1 shows the international links terminated at APAN Tokyo XP in 2002. SINET terminating Tokyo-Thailand link and AI3 [BIY01] interconnected with Tokyo XP. Figure 2.1 shows the network configuration of international link in APAN in 2002.

Each link is maintained by a testbed project funded by single or multiple countries, such as AIII project for Seoul-Tokyo and Seoul-Singapore links (APII links), TransPAC project for Tokyo-Seattle and Tokyo-Chicago links (TransPAC links), and etc. The testbed projects have their own acceptable usage policies (AUPs) along the project goal for the related link, as follow.

- TransPAC links (Tokyo-Seatle and Tokyo Chicago links)
  Research institutions permitted to use vBNS, Abilene, StarLIHGT2 and
  IMnet and those having collaborative researches with them

Table 2.1: International links in APAN

| Link | Owner | Bandwidth |
|---|---|---|
| Tokyo-Seatle | Indiana University | 622 Mbit/s |
| Tokyo-Chicago | Japan Science and Technology Agency | 622 Mbit/s |
| Tokyo-Beijing | Tsinghua University | 10 Mbit/s |
| Tokyo-Seoul | CRL[1] and KISDI[kis] | 8 Mbit/s |
| Tokyo-Philippine | MAFFIN | 768 Kbit/s |
| Tokyo-Malaysia | NTT Communications co. | 192 Kbit/s |
| Tokyo-Thailand | National Institute of Informatics | 2 Mbits/s |
| Fujisawa-AI3[*1] | JSAT | [*2] |
| Nara-AI3[*3] | JSAT | [*2] |
| Seoul-Chicago | KOREN[kor] | 45 Mbit/s |
| Seoul-Singapore | KISDI/ SingAREN[NYS+98] | 2 Mbit/s |
| Singapore-Seatle | SingAREN | 27 Mbit/s |

[*1]: Philippine, Singapore, Penang, Colombo, Hanoi

[*2]: Downlink from Japan: 1.5 Mbit/s、Uplink: 512 Kbit/s

[*3]: Hong Kong, Bandhan

- APII link (Tokyo-Seoul link)
  Research institutions making a contract under the framework of the APII project. For the ease of network operations, the permitted researchs institutions to use APII links are unified with TransPAC links.
- Tokyo-Philippine link
  All the research institutions.
- Tokyo-Malaysia link
  All the institutions using IPv6 communication.
- Tokyo-Beijing link

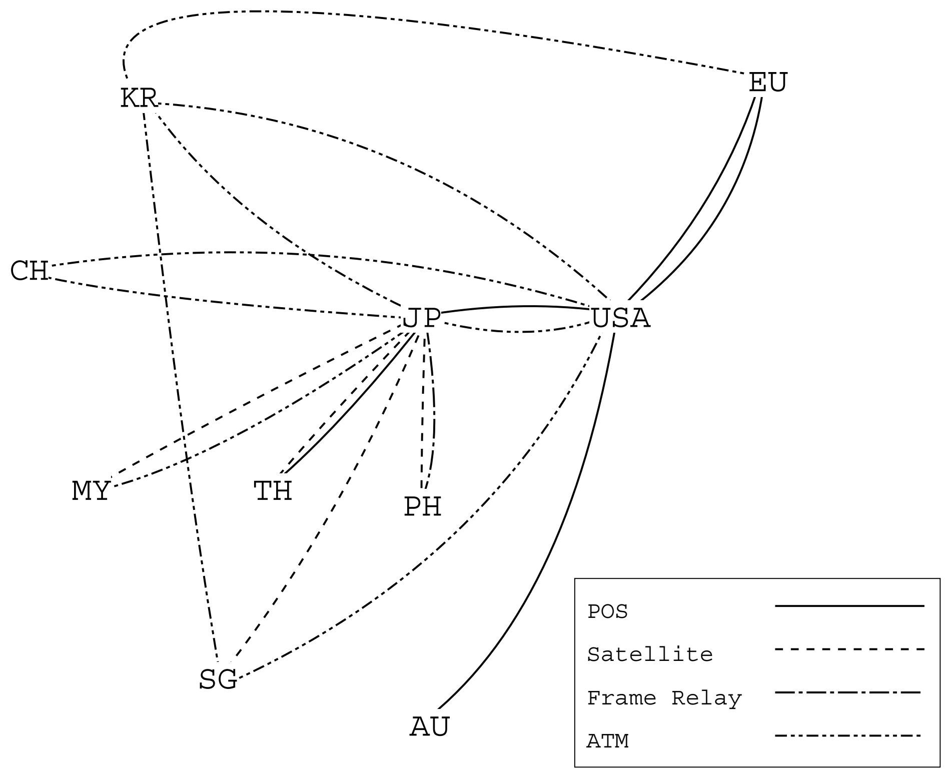


Figure 2.1: International links in APAN

All the institutions communicating with (Chinese) research institutions connected to CERNET.

APAN introduced a project-oriented user restriction reflecting the AUP in the IPv4 transit service. while there is no user restriction for IPv6 and multicast services. The IPv4 AUP limits the communication to only those between users registered with a research project accepted by link owner. That is, users *A* and *B* in a research project *p*, and users *X* and *Y* in a research project *q* can communicate between *A* and *B* and between *X* and *Y*, while the communications between *A* or *B* and *X* or *Y* are prohibited.

The AUPs of TransPAC and APII links results in forcing routers to examine the source address in forwarding traffic. Hereafter, an institution permitted to use TransPAC or APII links is termed an *authorized institution* for the link.

APAN calls for a research project using the single or multiple links instead of each of link owers. APAN announces the subscribed project to related link owners, and then they examine if the subscribed project is acceptable for the goal of their testbed project. This process mitigates the complexity of the simultaneous project subscription to multiple testbed projects for the subscribers.

### 2.1.2 Network configuration

Figure 2.2 shows the physical network configuration of Tokyo XP. Leyer 2/3 switches (Foundry BigIron 4000) are used for connections between routers and between routers and servers in Tokyo XP. The most connections to the RENs are terminated at ATM switches (Fore ASX-200BX). and SONET/SDH, satellite, and frame relay are used for TransPAC, AI3, and Tokyo-Philippine links, respectively. The main routers 1 and 2 peer with 24 RENs or authorized institutions and exchange traffic coming from them mostly via ATM permanent virtual circuits (PVCs). The use of double main routers (Juniper M20) is for redundancy. So, the authorized institutions and the RENs having only the authorized institutions as thier users establish ATM PVC and BGP [RL95] peering session with both main routers 1 and 2. The links from those RENs that forward traffic from both the authorized and the non-authorized institutions to Tokyo XP, are terminated at a policy router (CISCO 7505) distinguishing the traffic of the authorized institution from the mixed traffic. In addition, Tokyo XP equips IPv6, Diffserv, and multicast routers for supporing users launching various experiments.

### 2.1.3 Network Operations

The network operations in Tokyo XP, are achieved by the collaboration of designing, operatoring, monitoring of the network. Designing Tokyo XP requires understanding a whole network configuration in APAN, the connections to peering RENs and research projects, and

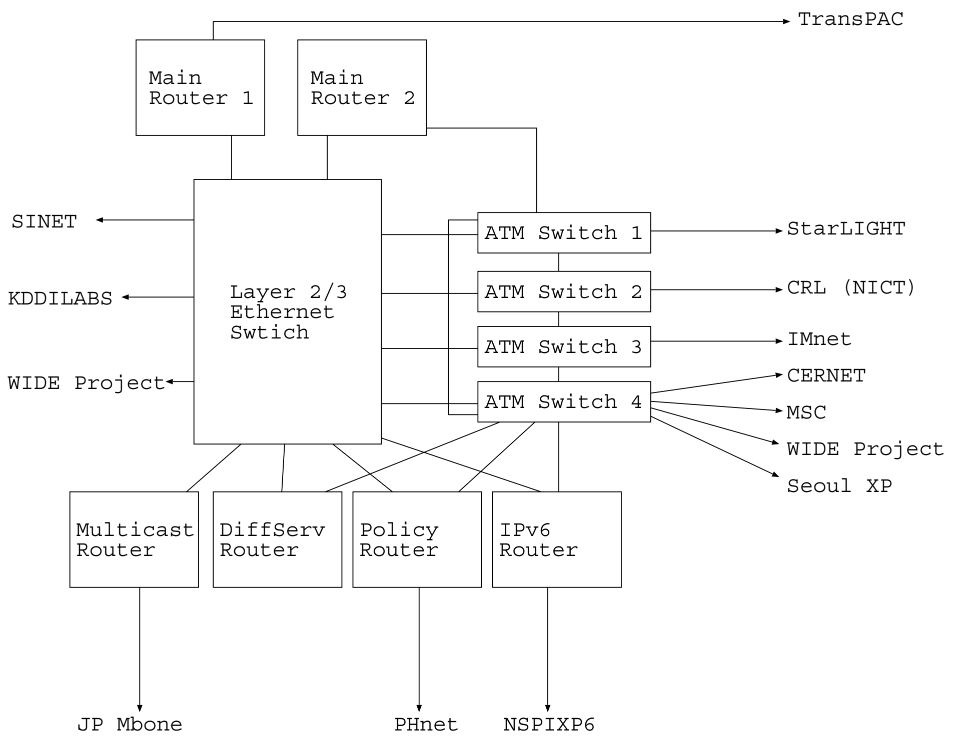


Figure 2.2: Physical network configuration

AUPs for individual links. Changing network configuration is conducted by operators in practice along the designer's plan. To reducing operational cost, various monitorings, such as, traffic visualization, the reachability to routers and servers, and BGP peering status, are automatically conducted.

### 2.1.4 Monitoring Tools

Every XP in APAN employs various network monitoring tools. Most of them are freely available software while some are originally developed in XPs, as follow:

- *Traffic viewer with CoralReef* [KMK$^+$01] [MKK$^+$] visualizes time-varying traffic traversing ATM links with higher resolution, e.g., the order of seconds, than Multi Router Traffic Grapher (MRTG) [mrt] which is the freely available and most famous and publicly-used traffic viewers

- *BGP-route table viewer* shows the BGP routes arranged by the BGP peers in WEB page.
- *Reachability checker* measures the statistics of round trip time, and the rate of packet loss to multiple nodes simultaneously.
- *BGP session checker* polls BGP peering status information via SNMP and report the status to operators if the status changes as the operator defines.

Any information collected from above mentioned freely available tools is publicly opened on the WEB of Tokyo XP. and which results in helping the peering RENs to segregate their problem where a problem happens and to troubleshoot the problem by referring the information.

### 2.1.5 Routing

### 2.1.6 IPv4 Unicast

Tokyo XP uses OSPF for the interior routing protocol (IRP), and BGP4+ [BRCK00] for the exterior routing protocol (ERP). BGP4+ has the ability separately to maintain multicast and IPv6 routes as well as IPv4 unicast. The IPv4 unicast routing is conditioned by the follow

1. Traffic sent to or from XPs and RENs themselves are forwarded without any restriction, because of recognition such a traffic as the network operation use.
2. Traffic forwarded to the authorized institutions is examined its source address as well as the destination address. The examination of the destination address is performed by refering the routing table for the authorized institutions as usual.

3. In the other case, traffic that its destination is those networks announced by the peering RENs is forwarded to the RENs. However, in this case, TransPAC and APII links are not used.

The condition 1 is exceptionally permitted such a communication among RENs due to is vital for the troubleshooting in the abnormal situation, or for the verification of changes.

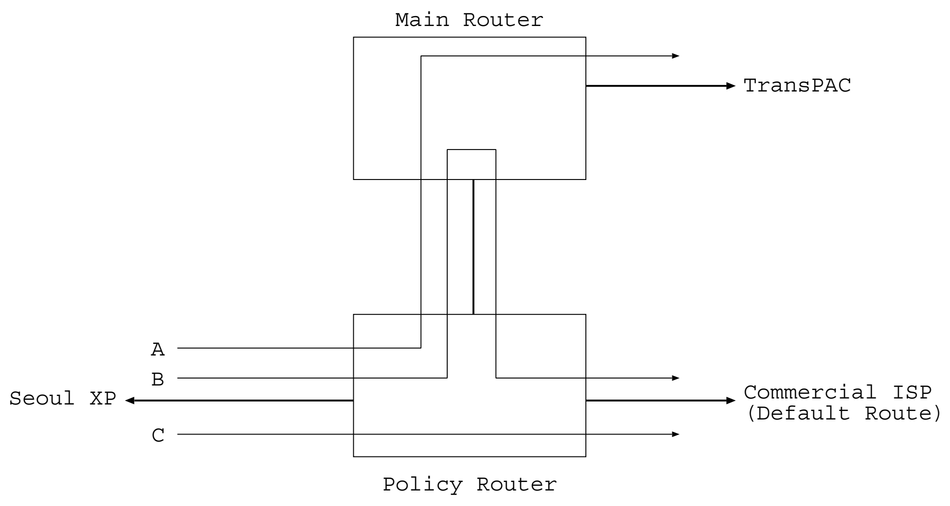
Examining the source address of packets (condition 2) or other fields of IP or transport header besides destination address is known as the PBR. In Tokyo XP case, first the source IP address of incoming traffic are examined, and then the routing table for the authorized or the non-authorized institutions is selected. Because all the routers in Tokyo XP maintained only the single routing table, Tokyo XP introduced the policy router to examine the source IP address of each packet in order to segregate traffic generated by the authorized institutions from traffic mix. Some ASs connecting to Tokyo XP send traffic from both the authorized and the non-authorized institutions for TransPAC or APII projects. This is because the authorized institutions for TransPAC or APII projects are decided by the project owner, without regard to the transit RENs which the institution selects to reach to Tokyo XP.

Figure 2.3 shows the multiple router scheme for the PBR. The paths along which packets are forwarded in Tokyo XP differ on the source address of the packets. When there is no routing information for a packet forwarded to the main router, the packet is forwarded to the policy router which has the default route to the outside of Tokyo XP.

The PBR is conducted with setting the map of the condition, such as, source IP addresses, and next-hop to which packets are forwarded, which is given by the function from the CISCO router.

Because only the single next-hop is configurable in CISCO router, we emply the virtual router redundancy protocol (VRRP) [KWW+98] allowing multiple routers to share the single IP address for the next-hop address.

.



| | Source Address | Destination Address |
|---|---|---|
| A | Authorized Institutions | Authorized Institutions |
| B | Authorized Institutions | Non-authorized Institutions |
| C | Non-authorized Institutions | Non-authorized Institutions |

Figure 2.3: Policy-based routing scheme

### 2.1.7 IPv4 Multicast

Tokyo XP has a role of JP Mbone border. In Tokyo XP, protocol indipendent multicast for sparse mode (PIM-SM) [EFH+98] is used for the multicast tree construction. Multicast source discovery protocol (MSDP) [FM03] enables Tokyo XP to share multicast sources among ASs The routing information exchanged for the unicast of the internal Tokyo XP by OSPF is exploited for the multicast, while BGP4+ exchanges external routing information in order to have the exclusive routing table for multicast. Having the different routing tables between unicast and multicast for the external routes is vital, because almost all ISPs and RENs refrain from providing multicast services in the current Internet. In addition, it solves the difference of AUPs between unicast and multicast in the Tokyo XP Multicast source discovery protocol collects the source address information of the on-going multicast.

The RENs and research institutions without direct link to Tokyo XP can establish tunnel link over which multicast related protocols are used. Although conventionally, DVMRP tunnels are mostly used, currentl, IP in IP tunnel or generic routing encapsulation (GRE) tunnels are widely used.

### 2.1.8 IPv6

APAN has actively been constructing dual stack networks for both IPv4 and IPv6 since the early phase of IPv6 deployment in the world. Communication using IPv6 is not restricted over any link at all as well as that of multicast, because the IPv6 is still experimental phase. Therefore, IPv6 BGP peering and communications are unlimited even the peering is with ISPs.

OSPF2 and BGP4+ are used for the IRP and ERP, respectively. In a similar way, the research institutions and RENs without any direct link, can establish tunnel links over which IPv6 related protocols are used.

## 2.2 Evaluation of Policy-base Routing Scheme

In general, a router product is optimized for routing table look-up with the packet destination address in packet forwarding. To verify the policy based routing scheme which forces a router to examine the source IP address in forwarding, the packet forwarding performance including source IP address checking was evaluated.

### 2.2.1 Experimental Environment

Figure 2.4 shows the network configuration for the evaluation. A policy router and a main router are directly connected with an OC3 ATM link. An ethernet switch logically connects each of the routers with a PC running FreeBSD version 4.5 with port VLAN. A packet sent from one PC thus reaches to the other via both routers. 100BASE-TX (full duplex) and

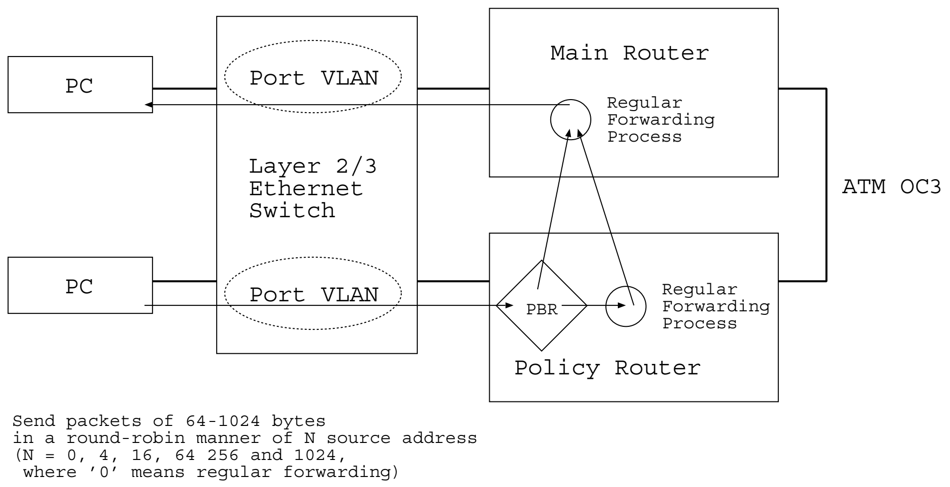

PC
PC
Port VLAN
Port VLAN
Layer 2/3
Ethernet
Switch
Main Router
Regular
Forwarding
Process
ATM OC3
PBR
Regular
Forwarding
Process
Policy Router
Send packets of 64-1024 bytes
in a round-robin manner of N source address
(N = 0, 4, 16, 64 256 and 1024,
where '0' means regular forwarding)

Figure 2.4: Network configuration for the PBR performance evaluation

1000Base-SX are used for links between the PCs and the switch, and between the routers and switch, respectively. The CISCO 7505 equiping Virsatile Interface Processor (VIP) [vip], Juniper M20 equiping Internet Processor II, and Foundry BigIron 4000 equiping Management Module IV are used for the policy router, the main router, and the ethernet switch, respectively.

In the experiment, the PC connected to the policy router (source PC) generated packets towards the other PC (destination PC). The policy router examined source IP address of the packet, and directly forwarded the packet to the main router if it matched with the registered IP address, or processed the destination address-based table look-up before forwarding the packet to the main router, otherwise. The maximum packet rate with no packet loss in the policy router was found as the packet forwarding performance of the policy router. The evaluation was conducted for 64, 256, and 1024 bytes of generated packets, and for 4, 16, 64, 256, 1024 source IP address entries of PBR setting. The source IP address of generated packets was varied in a round-robin manner in the double range of the PBR entries (from 8 to 2048) in order for every two packet evenly to match with all the PBR entries. The packet rate was derivated from the averge of ten examinations for the individual conditions.

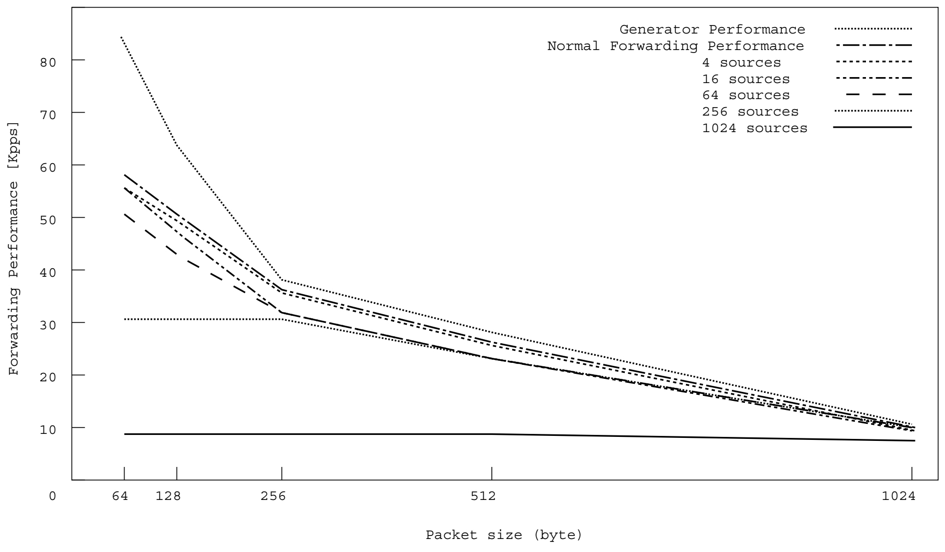


Figure 2.5: Packet forwarding performance

### 2.2.2 Packet Forwarding Performance

Figure 2.5 shows the packet forwarding performance when varying the number of source IP address entries. “Generator Performance” indicates the performance when the two PCs are connected via only the ethernet switch. “Normal Forwarding Performance” indicates the performance when all the packets are forwarded with the regular destination address-based table look-up. The horizontal parts for 1024 bytes of packet size is due to the wire rate limitation of 100 Mbit/s link (12 Kpps). Those for 64 and 256 bytes or packet are probably because there exist bottleneck related to the regular destination address-based packet forwarding. In case or 128 or larger number of PBR entries, the load of checking the source IP address increasingly become heavy resulting in affecting the forwarding performance.

VIP in CICO7500 series has an exclusive processor for the regular destination address-based packet forwarding performed without routing processor (RP) of the router for only the single or double network interface cards, However, the PBR process was performed by the RP in the experiments. In the cases of the large number of PBR entries (256 or more), the load of RP reached to about 100 % resulting in the the performance degradation.

Considering the required forwarding performance of PBR in Tokyo XP, the actual traffic volume in Tokyo XP requiring the PBR is up to 12.5 Mbit/s including 10 Mbit/s for CERNET link (China), 2 Mbit/s for NECTEC link (Thailand), and 512 Kbit/s for AI3 links, that is, 24.4 Kpps, 6.1 Kpps, or 1.5 Kpps in cases of 64 bytes, 256 bytes, or 1024 bytes of packet sizes, respectively. The actual number of PBR entries is only 19. Hence, the evaluation proves that the PBR scheme can afford the required forwarding performance in Tokyo XP

## 2.3 Consideration of Backbone Network Architecture

The PBR, that is, segregating traffic to forward along the different paths with some fields in IP and/or transport-layer headers besides destination IP address, is vital to manage different AUPs in the single network, as described in this chapter. Increasing the capacity of accepting traffic and providing better QoS require the optimal traffic distribution in the network. The fundamental philosophy for making the Internet scalable is that algorithmically complex processing should be pushed to the edge of the network whenever possible. In terms of the future bakkbone network expected to deal with hundreds-Gbit/s traffic, traffic segregation and traffic distribution should be processed at the edge routers of the backbone network, while routers in a core network of the backbone network should be exclusively designed to forward traffic as large as possible.

The effective traffic distribution can be achieved by classifying traffic into flows, balancing the individual flows over multiple paths with the traffic characteristics, and routing multiple paths including re-assigning bandwidth to the individual paths in the core network between the individual pairs of edge routers [EJLW01] [GKL$^+$04] [SGD03]. The flow classification requires the traffic measurement on each flow for finding its burstiness with which flows are effectively assigned to multiple paths. The functions required in flow-based traffic measurement are described in the development of the real-time IP-flows monitoring tool in Capter 3. The effectiveness of employing the traffic characteristics in distributing traffic is proven in Capters 4 and 6.

The edge router can solely measure the traffic characterisitcs of flows and distribute the flows over multiple paths, while optimally routing paths in the core network is achieved by the cooperations of the edge routers and nodes in the core network. Therefore, the parameter setting of flow assignment processed in an edge router according with the time-varying traffic is reconfigured with a shorter interval than that of routing paths in the core network. Routing paths increases the capacity of admitted traffic and improves its QoS again when the traffic varies largely which can not be addressed by the flow assignment among fixed-routed and fixed-bandwidth paths.

This dissertation focuses the flow-based traffic measurement and flow assignment achieved in a edge router when multiple fixed-routed and fixed-bandwidth bandwidth paths are given. Thus, Exploiting the traffic characteristics is effective in the traffic engineering especially improving the addmitted traffic and the QoS for the traffic. Routing paths among all the pair of routers over core network as a whole is remained as the future work.

## 2.4 Conclusion

In order to create new communication services, to develop high performance communication to improve QoS, the high performance research and education networks (RENs) emerge to be build in the world. Asia-Pacific Advanced Network (APAN) provides various services for research and educational uses in Asia-Pacific region. Since APAN is a multipoint RENs composed of multiple point-to-point links individually owned various institutions in Asia-Pacific region, such a formation forces the exchange points (XPs) of those links to be responsible for acceptable usage policies (AUPs) of individual links in forwarding traffic. Concretely, XPs are required to segregate traffic based on the pair of source and destination address and exploit the routing table referring the specific AUP based on the segregated communication in forwarding traffic.

Tokyo XP which is the biggest one in APAN, adopts policy-based routing (PBR) correctly to select the routing tables in forwarding traffic. Since PBR enables router to segregate traffic with not only destination IP address but also all the fields in IP and transport layers, such a complex process is not as optimized as next-hop search with the destination IP address. In addition, routers in Tokyo XP maintain the single routing table. Therefore, Tokyo XP employs the multiple router scheme for the PBR in order to improve the process performance and maintaining multiple routing tables.

In this chapter, the summary of the required services, the network configuration and routing providing the services, various tools employed for reliable operation, and PBR scheme using multiple routers with the single routing table in Tokyo XP were described. And the evaluation of the PBR scheme proved that the scheme has the sufficient ability to deal with the traffic volume required in Tokyo XP. In addition, we considered the architecture and the required functions in backbone network dealing with some hundreds Gbit/s traffic with various types of traffic for improving the capacity of admitted traffic and QoS.

In Chapter 3, the required functions of flow-based traffic measurement, which is one of the required functions are described, are clarified through the developing the real-time IP-Flow monitoring tool. And the effect of the flow assignment exploiting the traffic characteristics of flows, which is another function required in the backbone network, are proved through the traffic analysis of the distributed computing applications in Chapter 4 and through developing the flow assignment method in Chapter 6.

# Chapter 3

# Development of Real-time IP-Flow Monitoring Tool with Scalable Architecture

## 3.1 Introduction

There is an emerging requirement for real-time flow-based traffic monitoring, which is vital to detecting and/or tracing DoS attacks as well as troubleshooting and traffic engineering in the ISP networks, instead of the existing IP-layer traffic volume monitoring or off-line flow analysis of collected traffic data. For example, fine-grained and user-defined flow monitoring is of practical importance for performance sensitive services such as Grid applications, while such monitoring on very high-speed links is a challenging task due to the large overheads to investigate the contents of every packet passing through the monitoring point.

The contents of traffic passing through Internet Service Providers (ISPs) are becoming diverse since various applications such as peer-to-peer, VoIP and so on, are widely used and DoS attacks occur frequently in their networks. It is getting more difficult in such networks to monitor the traffic of these applications as well as to detect and/or to trace DoS attacks with tools showing graphs of the whole IP layer traffic. Monitoring in such environments is required to classify traffic into flows.

On the other hand, the MPLS based traffic engineering requires monitoring flows to optimize usage of an entire administrated network. Managed networks such as Grid Computing research networks are expected to progress the optimization of flow controls based on both operational administrative policies and QoS requirements. A key issue is to detect what hinders a flow from reaching its target throughput. This kind of monitoring requires highly accurate flow measurement up to microsecond resolutions to understand how much bandwidth a flow consumes.

It is much more useful that flows can be specified by any field from the IP header up to application data in the payload as network operators require. Especially in the ISP operations, the following functions are very useful to understand the characteristics of problems and to reduce troubleshooting time: 1) extracting traffic flows specified by the operators on demand in a flexibly and impromptu manner; 2) visualizing them in a real-time manner. However, the existing tools for such flow measurements have several limitations and drawbacks as mentioned in Section 3.2. In general, the existing software-based systems are suitable for relatively slow links, while the existing hardware-based systems cannot achieve sufficient flexibility.

In this chapter, the architecture for a scalable real-time flow measurement tool is proposed. The tool allows operators to flexibly define “the targeted flows” on-demand, to obtain various statistics on those flows, and to visualize them in a real-time manner. The system implemented based on the proposed architecture consists of the multiple capture devices, the manager device and the user interface devices. A bit-pattern-based flow definition method and its data structure to measure multiple flows with flexible flow definitions are proposed. Finally the evaluation shows that the proposed system using six capture devices performs to measure flows with up to 80K pps traffic in case of giving multiple flow definitions.

## 3.2 Related Works

MRTG is a tool for collecting Management Information Base (MIB) information (typically byte counters of router interfaces every five minutes) from remote network devices by using

Simple Network Management Protocol (SNMP) and for visualizing time-varying characteristics of the information. MRTG is widely adopted in IP network operations because it is easy to use and automatically generates visual graphs and their HTML pages. MRTG theoretically can visualize flow-based traffic information based on RMON2-MIB [Wal00] in cooperation with RMON2 enable devices. However, there are several limitations: e.g., RMON2-MIB is not so flexible, RMON2 enabling devices are not so common, and collecting information cannot be performed in a short time interval due to the architectural limitation of SNMP.

NetFlow [net02] provided by Cisco or Cflowd [McR] developed by CAIDA [cai] collect flow statistics generated by sampling packets passing through the router. Real-Time Flow Measurement Working Group of IETF suggests that per-flow basis counters kept in router to export statistics to collectors. The keeping per-flow counts consume considerable memory as well as processing power if the number of flows becomes large.

sFlow [PPM01], which is also discussed in Network Working Group of IETF, is a specification to export raw data from sampling the traffic arriving at the switch besides the statistics. Some MIB of its statistics are shared with NetFlow MIB. The basic behavior of sflow export is that the sampled packets are chopped into a certain length and sent to multiple collectors. Per-flow network traffic measurement with sampled traffic has a trade-off between the sampling rate and the accuracy of measurement result as showed in the literature [DLT01] [DLT02]. Some routers or switches have a function to capture every packet passing an interface, however such processes often cause performance degradation in the packet forwarding process, and thus, are adopted only to slow speed interfaces.

Anritsu Cooperation provides hardware-based traffic monitoring tools which measures flows passing through a gigabit-ethernet link [anra]. Tools can store measured flows at millisecond resolution timestamps and their playback function generates traffic as captured. The system limits to efficiently measuring up to four flows at the same time.

In this chapter, the requirements of the flow measurement tool is discussed especially for network operation in ISP And, an architecture with scalability to meet high speed traffic and flexible flow definitions is proposed.

## 3.3 Requirements in Network Operations

### 3.3.1 Flow Measurement

Flow is defined as a set of packets passing an observation point in a network during a certain time interval and generally having the same 5-element-tuple of source IP address, destination IP address, protocol, source port number and destination port number [QZCZ04]. However, a flow is taken in a wider sense to define as a set of packets having common properties specified by not only fields in the header but also application data in the payload.

Flow measurement is defined as clarifying the traffic properties derived from traffic changes, statistical length (total number of packets or bytes) and existing time (interval between first and the last packets) of flows. The flow measurement process consists of capturing traffic, flow identification, statistic processing and data preservation which described in Section 3.4. The flow identification process requires high performance and often expensive hardware for high speed links.

### 3.3.2 Requirements

In ISP operations, The traffic monitoring while classifying traffic based on applications is required to meet DoS attack, P2P utilization limitation and so on. The monitoring is required an ability to distinguish a certain application traffic from various ones. However some of applications cannot be distinguished based on port number of transport protocol. Therefore application data fields in a packet are also used in that case.

Additionally, ISPs are required to monitor the quality of traffic for customers who make a SLA contract with them. The ISP operations are required abilities to detect and monitor the precise change of traffic to troubleshoot and to analyze the depression of such customer's traffic performance regardless of the traffic volume. The followings are the requirements for flow measurement to support such ISP operations:

1. *Processing Scalability*: The measurement system should be extensible in terms of its packet processing power. Flow identification processes require processing power as traffic increase.

2. *Flexible Flow Definition*: To diagnose and troubleshoot the customer traffic, the ability to define a flow regardless its volume is important. The flexible flow definition by specifying a flow with application data fields in packets leads to smooth operations in the recent Internet carrying various application traffic.

3. *Operational Flexibility*: To support multiple users to obtain various statistics for flows, the system should be able to accept to update the flow definitions on-demand and, to visualize them in a real-time manner. Additionally these operations are done by one interface.

4. *Long-term Operation*: System should be able to keep working as long as possible. Any replacement such as storage to meet the limited room, updates of system configuration and parameters.

5. *Flexibility of Visualization Resolution*: The system should give operators a spatially fine-grained view such as traffic volume per routes or applications, and a temporally fine-grained view such as millisecond-order traffic behavior, which exposes the burst of traffic which looks flat rate under the low (coarse) resolution. Additionally the ability to change such resolutions as operators required is useful for the operation in troubleshooting.

The scalability for real-time flow measurement system is defined as to realize all of these requirements.

Table 3.1: Comparison of requirements enabled by different measurement systems

| | NetFlow | sFlow | NeTraMet | Anritsu | proposed system |
|---|---|---|---|---|---|
| Processing scalability | ○ | ○ | | ● | ● |
| Operational flexibility | ○ | ● | ● | ○ | ● |
| Flexible flow definition | | | ○ | ○ | ● |
| Long-term operation | ● | ● | ● | | ● |
| Resolution flexibility | | ● | | ● | ● |

| | |
|---|---|
| ● | enabled |
| ○ | limited |

### 3.3.3 Issues

In this section, existing flow measurement system or techniques are surveyed to compare the previously clarified requirements of them.

NetFlow consumes memory and make main processor overwork in routers as the increase of interface speed. Therefore sampling technology is recently deployed in NetFlow to follow such a high speed. Although some collectors such as FlowScan [Plo00] provide multiple flow definitions, NetFlow running background of collectors exports all flow information and its overhead is not small between collectors and routers.

sFlow forwards sampled packets to collectors. The sampling rate is enlarged to follow high speed link. Although some collectors such as InMon Traffic Server [inm] provide multiple flow definitions, it's hard to detect flows having small traffic volume compared to the sampling rate.

NeTraMet provides the flexible flow definitions by any combination of addresses or ports of Datalink, Network and Transport layer. The collection of measurement information is done with SNMP. Therefore it cannot be performed in a short time interval due to the architectural limitation of SNMP.

The hardware-based traffic monitoring tools from Anritsu Cooperation support various link speeds by replacing the network interface cards. The system limits to measure up to four flows at the same time and the high resolution visualization is done in the off-line manner.

Table 3.1 summarizes existing systems and the requirements realized by them. The proposed system is discussed in Section 3.7.1.

In the ISP operations, multiple tools are used because any system cannot provide the all requirements previously described. It is effective that a real-time flow measurement tool supporting all requirements is deployed in ISP operations.

## 3.4 Proposed System

Generally, flow measurement system consists of the following components.

- Packet Capture Component
- Flow Identification Component
- Status Preservation Component
- Analysis Component
- Data Preservation Component
- User Service Component
- User Interface Component

In order to realize the scalability discussed in Section 3.3.2, the role of each component described in Section 3.4.1 and the architecture combining these components are proposed.

### 3.4.1 Enhancement of Component

The way to enhance each of the components to meet the scalability is summarized as follows:

- *Packet Capture Component*: Copying traffic from Network device is done with the mirror function provided high-end Ethernet switches or network taps. The optical tap can follow the increase of link speed due to no conversion between optic and electricity. The system should have a buffer for input traffic to absorb burst input. The system should also have the mechanism which independently handles process buffering input traffic and post-process. To prevent to make the timestamp inaccurate, buffering should be done after obtaining the timestamp in a packet arrival.

- *Flow Identification Component*: This component examines every packet matching with the flow definitions. Generally, the burden of flow identification process increases as the increase of traffic rate and the number of flow identifications. Both are not avoidable to support the Processing Scalability and the Flexible Flow Definition. The improvement the process performance (e.g. by hardware implementation) is worried to cost enormously. As the other solution, deployment of multiple non-high-end devices to distribute burden of flow identifications process is expected lower cost such as in proportion to the traffic rate or so.
- *Status Preservation Component*: This component checks the flow status by inactivity timeout which defined by user.
- *Analysis Component*: This component calculates statistics as the way of the each measurement attribute described in Section 3.4.4. The calculation is done based on the flow granularity defined on-demand to meet the Flexibility of Visualization Resolution. The statistics are sent to Data Preservation Component.
- *Data Preservation Component*: As the increase of the number of flows, the system is required to meet the limitation of storage for measured data. The system should place the measured data in multiple places and change the place when the total amount of data stored in a place reaches a certain threshold which given by a user. Additionally the system should provide

the information where the system currently accesses and history of its accesses. This enables recognize which data is safe to move.

- *User Service Component*: This component has two features. One is to update flow definitions used in the Status Preservation Component, the Analysis Component and the Data Preservation Component. The flow definitions are updated form the User Interface Component. The other is to send stored measurement data to the one of User Interface Components. The User Service Component manages the multiple accesses from User Interface Components and should prevent that the flow definitions become inconsistent.
- *User Interface Component*: This component communicates with the User Service Component to exchange flow definitions and measured data, and visualizes graph in various resolutions requested by operators. Traffic graphs are automatically updated by periodically collecting measurement data.

### 3.4.2 Proposed Architecture

The architecture that pipelines the process of components and that lets Flow Identification Component distribute workload of flow identification process over multiple devices to have good scalability is proposed. For the distribution of workload, the distribution component is introduced, and the additional functions are summarized as follows:

- *Distribution Components*: This distributes monitoring traffic to the multiple Flow Identification Components. Distribution Component is a key that the system follows the input traffic rate. Therefore its process should be as simple as possible to be implemented on hardware.
- It’s not expected that the Distribution Component distributes traffic based on the flow definitions, which makes the distribution process complex.

Therefore the Flow Identification Components should maintain the all flow definitions as considering the Distribution Component adopts a round-robin distribution for its simplicity.

- The system should be able to add and drop the Flow Identification Components to follow the requirements of process power while the system is running. Both the Distribution Components and the Analysis Component detect an addition of the Flow Identification Components. The information of Flow definition should be shared between the Status Preservation Component and the Flow Identification Components. The Distribution Component should be able to detect to stop distribution to the Flow Identification Components which stops its process or is removed. This function meets the Processing Scalability and the Long-term Operation.
- The Status Preservation Component re-orders the measured data based on the data timestamp. The measurement data is asynchronously sent from the Flow Identification Components.

Figure 3.1 shows the combination of the components described in Section 3.4.1 and data flow through them to pipeline the measurement process. The basic flow of process is as follows:

1. Traffic is copied by an Ethernet switch or an optic tap and forwarded to the Distribution Component.
2. It then forwards packets to one of devices having the Packet Capture Component and the Flow Identification Component.
3. The Flow Identification Component updates measurement data if the packet matches with flow identifications.
4. The Flow Identification Component periodically send the local measurement data to the Status Preservation Component.

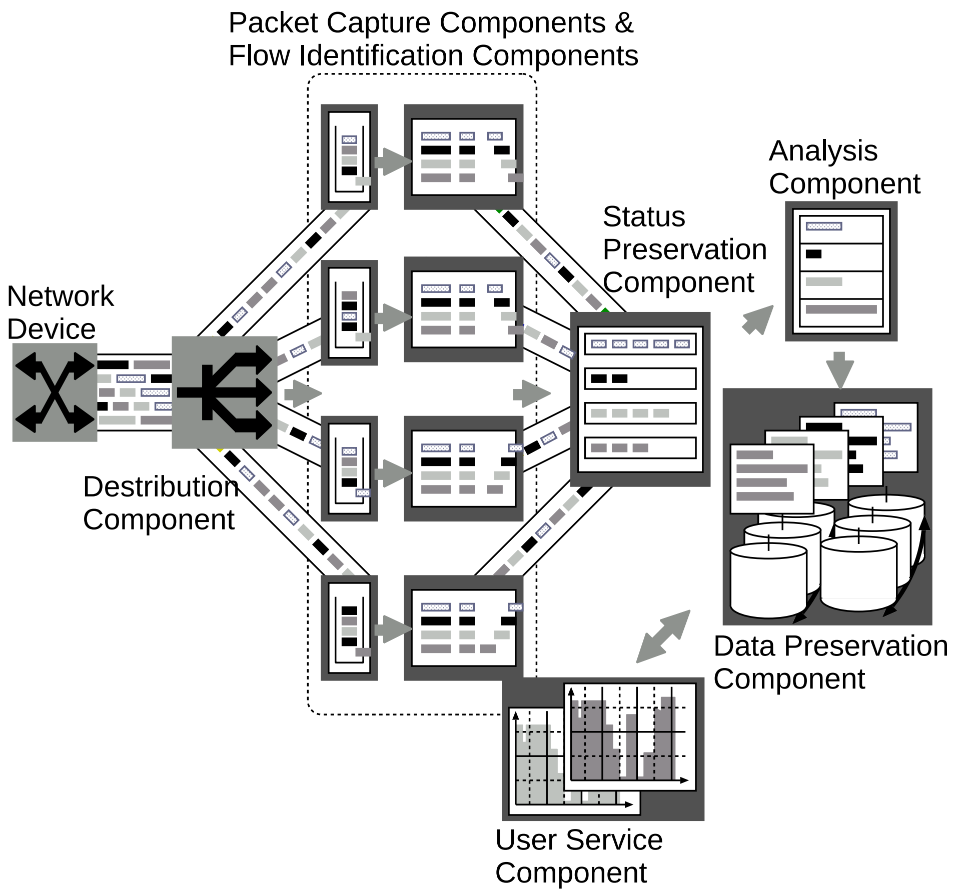


Figure 3.1: Architecture of a distributed real-time flow measurement tool

5. Partial measurement data from the Flow Identification Component is re-ordered and flow status is checked in the Status Preservation Component.
6. Statistics of measurement attribute calculated in the Analysis Component and it sent to Data Preservation Component.

This architecture enables the system to update flow definitions on-demand (the Operational Flexibility), to move the accumulated data and to add or drop the Flow Identification Components without stopping the system (the Long-term Operation), to accept the various measurement granularity of time-scale (the Flexibility of Visualization Resolution).

Table 3.2: Elements specifying the bit-pattern

| | |
|---|---|
| Pattern ID | Identifier used for reference by other bit-patterns |
| Position | Position of bit-pattern from a top of IP packet |
| Length | Length of bit-pattern |
| Mask | Specifies a valid and invalid bits |
| Minimum Pattern | Minimum value to specify range of bit-pattern |
| Maximum Pattern | Maximum value to specify range of bit-pattern |
| Child Pattern ID | Reference list of chained bit-patterns |

The Flexible Flow Definition is discussed in the following section.

### 3.4.3 Flow Definition and Data Structure

A flow definition is defined as a set of chained bit-patterns. The bit-pattern is defined by elements as described in Table 3.2. The ’chain’ of bit-pattern implies the ’AND’ operation for pattern matching.

When the capture device receives a packet, it picks up a bit field specified by parameters of a bit-pattern. This bit field is termed as a flow identifier (FID) hereafter. The original packet is judged as matching with the bit-pattern if its FID is between a range specified by the minimum and maximum pattern. The FID obtained from the last bit-patterns chained for a flow definition is used as a key to search and update measurement attribute described in Section 3.4.4. The combination of a flow definition and the last FID identifies a flow in the proposed architecture.

By taking a flow definition into multiple bit-patterns, the multiple flow definitions can be converted into hierarchical chained bit-patterns to eliminate duplicate pattern-matching per packet. Figure 3.2 shows an example of hierarchical chained bit-pattern for measuring WEB and NEWS traffic. Bit-pattern: “IPv4 TCP” is used when the packet is judged as IPv4 packet by bit-pattern: “IP version” which checks version field of IP header. Both “IPv4 TCP

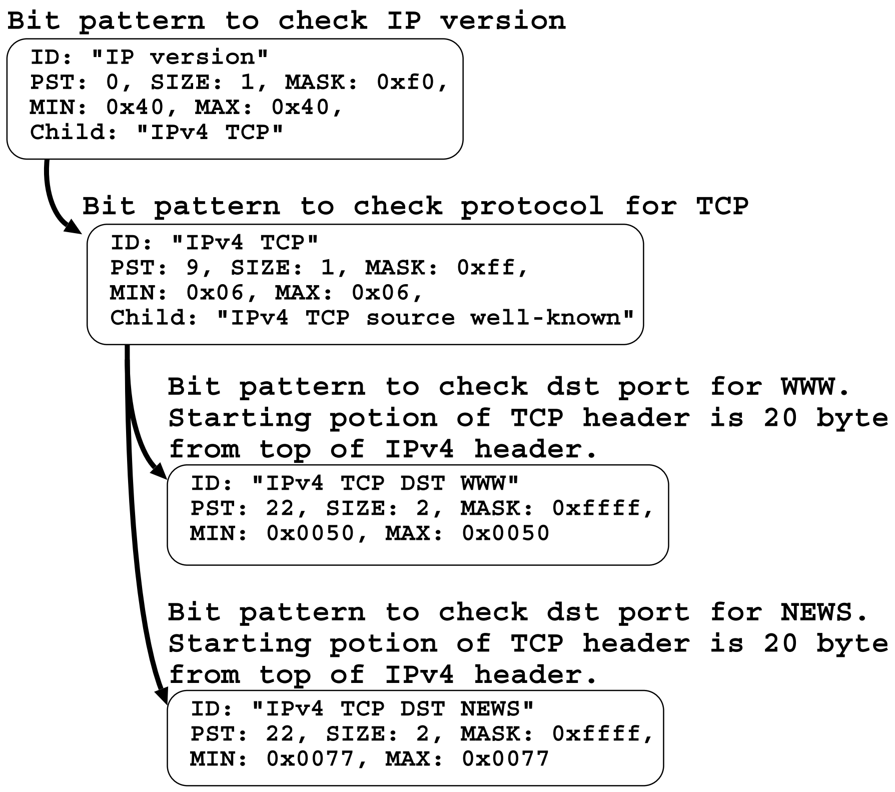

Bit pattern to check IP version
ID: "IP version"
PST: 0, SIZE: 1, MASK: 0xf0,
MIN: 0x40, MAX: 0x40,
Child: "IPv4 TCP"
Bit pattern to check protocol for TCP
ID: "IPv4 TCP"
PST: 9, SIZE: 1, MASK: 0xff,
MIN: 0x06, MAX: 0x06,
Child: "IPv4 TCP source well-known"
Bit pattern to check dst port for WWW. Starting potion of TCP header is 20 byte from top of IPv4 header.
ID: "IPv4 TCP DST WWW"
PST: 22, SIZE: 2, MASK: 0xffff,
MIN: 0x0050, MAX: 0x0050
Bit pattern to check dst port for NEWS. Starting potion of TCP header is 20 byte from top of IPv4 header.
ID: "IPv4 TCP DST NEWS"
PST: 22, SIZE: 2, MASK: 0xffff,
MIN: 0x0077, MAX: 0x0077

Figure 3.2: Example of hierarchical chained bit-patterns

DST WWW" and "IPv4 TCP DST NEWS" share "IPv4 TCP" and "IP version." Although the total number of bit-patterns from two flow definitions (WEB and NEWS) is six, that of hierarchical bit-patterns becomes four. For instance, when a UDP packet is examined with those bit-patterns, it doesn't match with "IPv4 TCP." Hence, the following pattern matches aren't done.

### 3.4.4 Measurement Attribute

Four measurement attributes for a flow definition are described as follows:

- *Flow Count:* Number of packets and bytes at certain intervals defined by users.

- *Flow Length:* Statistics on total number of bytes, packets and duration of a flow.
- *Packet Gap:* Statistics on timestamp intervals of consecutive packets composing a flow.
- *Association Packet Gap:* Statistics on timestamp intervals of two packets matching two different flow definitions.

When the Flow Count or the Flow Length are enabled, each of the multiple flows detected by a flow definition has its counters or length values. The Packet Gap is assorted the only flow definitions of exact pattern matching in which maximum and minimum patterns are the same. A flow definition can have multiple measurement attributes. The intervals for the Flow Counter, the Packet Gap, the Association Packet Gap and values of durations of the Flow Length have microsecond precision.

The Association Packet Gap calculates statistics from timestamp intervals of two packets that are expected to pass an observation point in order and that need two different flow definitions to detect respectively. E.g., a pair of packets having a flag SYN and FIN in the TCP header respectively to measure the duration of the TCP connection. The first flow definition is used to match the first packet , and then FIDs and timestamps are collected. The FID is substituted for the minimum and/or maximum patterns of the last bit-pattern for the second flow definition as users previously define. The second flow definition is generated with first packet FID and a lifetime whenever a packet matches with the first flow definition. Hereafter, the second flow definition begins to be used to detect the second packet expected to appear after the first one. The second flow definitions are released when a packet matches with it or its lifetime expires before a packet matches. After a packet matches with the second flow definition, timestamp intervals are computed from both the first and the second packets.

The user can limit the number of FID collected from the first flow definition to prevent the FIDs consuming resources in case that a long lifetime is given. When the number of FIDs reaches its limit, the following FIDs are discarded until the active FIDs are released.

## 3.5 Implementation Issues

In this section, the implementation issues are argued for implementing the system with the PC-UNIXs as the examples based on the proposed architecture, time accuracy, time synchronization and FID search.

### 3.5.1 Implementation with PC-UNIXs

In this subsection the implementation of proposed architecture with PC-UNIX is discussed. The following shows the composition and physical devices:

- *Distribution device*: As explained in Section 3.4, a hardware-based general-purpose distribution device is assumed as the Distribution device.
- *Capture device*: The Capture device consists of the Packet Capture Component and the Flow Identification Component.
- *Manager device*: The Manager device consists of the Status Preservation Component, Data Preservation Component and User Service Component.
- *User Interface device*: The User Interface device consists of the User Interface Component.

The capture device consists of three thread processes; Packet Buffering Process, Flow Identification Process, Report and Definition Update Process. Report and Definition Update Process periodically reports measurement data to the manager device and receives update of flow definitions. The Report and Definition Update Process derives or releases the Packet Buffering Process and the Flow Identification Process after that the capture device starts. All processes share flow definitions and measured flow information constructed in a memory space. For capturing packets, PCAP library [MJ93] was adopted it provides timestamp and an interface to collect capturing loss to application programs attempting to detect and count.

It is also supports both IPv4 and IPv6, and makes it easier to implement/port a capture device on/to the variety of OSs.

The manager device consists of four thread processes; Main Process, Definition Advertisement and Collection Process, Save Process, User Interface Service Process. Main Process manages the connections established from the capture devices or the user interface devices and derives other three processes. Definition Advertisement and Collection Process receives reports from the capture devices and advertises flow definitions when detecting any difference between registered definitions and definitions in a report. The Status Preservation Component and the Analysis Component are realized in this process.

Time synchronization is required between the multiple capture devices to prevent inconsistent reports between them. The accuracy of timestamp and method of time synchronization are discussed in the following sections.

### 3.5.2 Time Synchronization

In case that timestamp is given by the capture devices, the packet arrival time becomes inaccurate against the actual arrival time at the distribution device. The factors of this are differences of 1) packet forwarding delay and 2) process delay to take in packets in the capture device. The timestamping by the capture devices cannot avoid these factors even time synchronizes between capture devices completely.

1 can be minimized if the number of hop between the distribution device and the capture devices is few (e.g. 1 hop) and the connections between them consists of the same length cables and high speed device (e.g. gigabit-ethernet switch). Although 2 is caused by the burden of flow identification or other tasks running concurrently, the computer which exchanges gigabit class traffic is expected to have small delay. E.g. A computer which receives 1 Gbit/s (MTU: 1500B) takes in 12 microsecond per packet treatment in average. The roughly estimated accuracy is up to 1 millisecond, since the implementation is based on software in this proposal.

Above discussion is based on that time synchronizes between the capture devices. For the time synchronization, there are several clock sources are available as follows:

- *GPS*: GPS requires a sky view for an antenna. The establishment of antenna may be limited the building maintenance or security policies of data-centers where the system installed.
- *Atomic Fountain Clock*: The facility locations are limited. Wide area deployment is difficult.
- *CDMA*: CDMA (Code Division Multiple Access) is becoming a widely popular air interface for mobile telephony. The ntp (a free NTP server [Mil85]) supports CDMA as its clock source. CDMA brings better chance to deploy in building than GPS if its location is in the area of CDMA.
- *Clock Generator*: It's hard for the clock generator to deploy in wide area. The ntp also supports a few clock generation devices.

Note that each scheme can take both forms to receive clock signal from source and distribute PPS (pulse per second) to multiple receivers and then to computer, or to receive clock signal by receiver through an antenna and distribute PPS to multiple computers. In the implementation of proposed architecture, any clock source is available because capture devices should be installed in the same LAN segment or in few hop topologies to get better timestamp accuracy as discussed above. The selection of clock source is done with the other criteria, Wide area deployment, establishment of antenna and so on.

### 3.5.3 Flow ID Search

The range definition of a flow definition results detecting multiple flows in the system. Each flow matched with a flow definition is distinguished by the last FID of chained bit-patterns.

When a packet matches a flow definition, the same FID is searched from the previously registered FIDs. If the same FID does not exist at that moment, a new FID is registered. In case that many FIDs are registered due to large range size of bit-patterns, The FID search tends to take more time.

For searches with a certain length key such as the FID, a binary search can reduce the processing cost to $O(\log n)$ where $n$ is the total number of FIDs registered. However the Flow Identification Component registers FID and the search tree grows gradually thus the search tree must be well balanced. The AVL tree [AVL62], which keeps the search tree balanced, is one of solutions. However the insertion of a new FID is costly as opposed to normal list process, thus it is better to discover in what range size of bit-pattern the AVL tree shows better efficiency.

## 3.6 Evaluation

The first evaluation is if the system works as expected. The follows are confirmed.

- Flow definitions specifying any fields of a packet are accepted by the system and correctly match packets with appropriate flow definitions.
- The system accepts updates of flow definitions with any measurement attributes on-demand and hereafter it measures and stores statistics of requested attributes.
- The system can keep to run while the capture devices are added or dropped. The measurement data preservation function works correctly.
- Traffic graphs are showed on the user interface device and its auto update function works correctly.
- The system accepts multiple access from the user interface devices and prevents conflicts of flow definitions updates.

These evaluations imply that the system satisfies the requirements except the Processing Scalability. In the rest of this section, the performance and scalability are evaluated with the following experiments:

1. *Evaluation of capture device process performance when the Packet Buffer Process is used or not.* The distribution device asynchronously send packets to the capture device. The capture device is not ready to take a packet when it arrives, due to the previous packet process engaged. This evaluation proves how the packet buffering process is effective in the packet capturing process.

2. *Evaluation of the performance difference of capture device in deploying AVL Tree Search or the Sorted List Search.* This evaluation finds the performance difference between AVL Tree Search and Sorted List Search employed by a capture device The overhead of FID search may have an influence on the performance of capture device.

3. *Evaluation of how the entire system performance increase based on increment of the number of the capture devices.* This evaluation shows how the measurement performance improves as the number of capture devices increases.

### 3.6.1 Evaluation Environment

Two types of test, for the evaluations 1 and2, and for the evaluation 3 were conducted. The former examined the process performance of a capture device by measuring capturing loss which is caused by an overwork of processing. The traffic is sent with 500, 2K, 4K, 8K, 10K, 12K, 15K, 18K, 20K, 24K, 27K, 30K packets per second (pps) respectively from a traffic generator to a capture device in Figure 3.3 configuration. Flow definitions are registered in a manager device through a user interface device. The manager device advertises flow definitions to the capture device. The capture device makes Flow Counter reports for all FIDs to the manager device every 10 seconds. The performance is defined as the maximum speed in which there is no capturing loss for 60 seconds generation over 10 times examinations. The traffic generation is shaped by an application that authors developed to keep the short

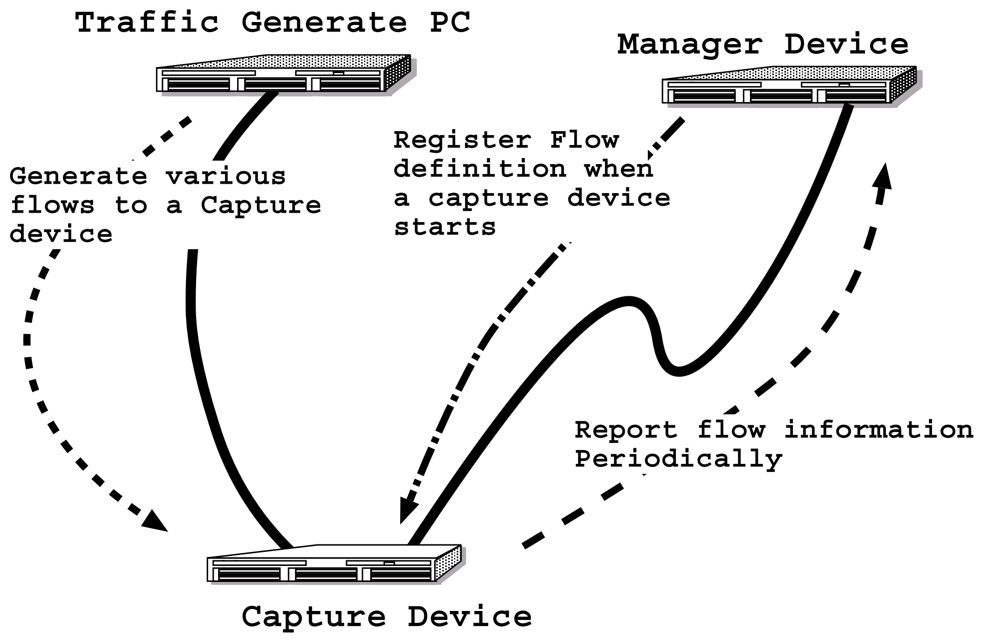

Traffic Generate PC
Manager Device
Register Flow definition when a capture device starts
Generate various flows to a Capture device
Report flow information Periodically
Capture Device

Figure 3.3: Configuration of performance evaluation

jitter at the user program level. However small burst traffic was observed during generation because the final packet treatment was controlled by a kernel.

The other evaluation is a performance test of the entire system for given 4, 8, 16 and 32 bit-patterns and the number of capture devices.

Figure 3.4 shows a configuration of the performance test. Although the configuration does not match with proposed architecture, each traffic generated from a generator is assumed as the traffic from distribution device in the practice. This can evaluate the entire system performance by the total the generated traffic. In this test, GPS was exploited for time synchronization clock source for all capture devices. The clock signal received at an antenna was divided up to 6 receivers and they provided plus per second to the corresponding capture devices.

The performance is defined as the maximum value of total packet speed generated by the generators for 60 seconds without any measurement loss. The maximum value was searched

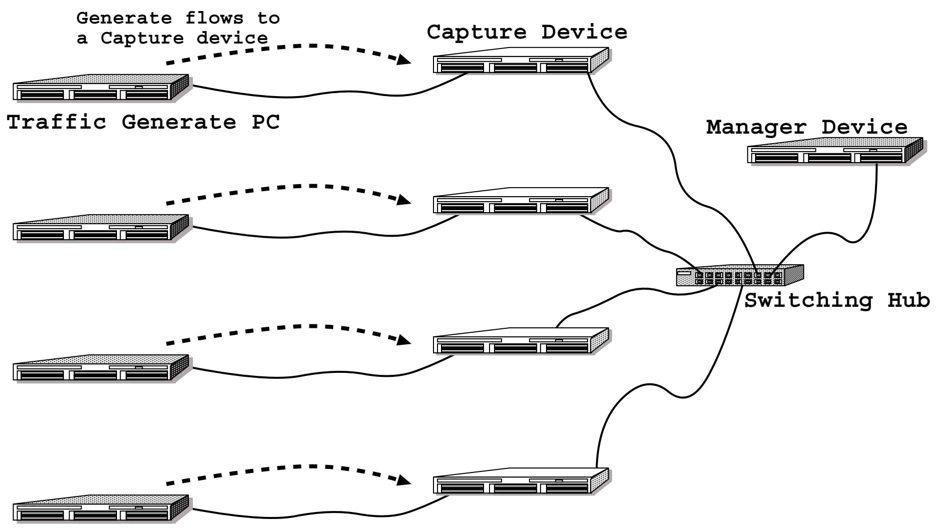


Figure 3.4: Configuration of entire system performance test

by binary search of packet speed. Bit-patterns were advertised to capture devices from the manager device every 10 second. Measured values of counter attribute were reported to the manager device every second. The precision of counter attribute was 10 milliseconds.

All computers in both evaluations had the same specifications as showed in Table 3.3.

### 3.6.2 Performance Difference in Deploying or Not Deploying the Packet Buffering Process

Figure 3.5 shows the process performance of capture device in which 1 up to 64 bit-patterns are registered respectively. Traffic which had bit-patterns to match with one of flow definitions was sent to the capture device. The bit-pattern appearance in generated packets was done in a round-robin manner. The length of a bit-pattern was four bytes and an exact match defined. This implies that the result is either matched or unmatched.

The maximum performance was 27K pps where the number of bit-patterns was 1 and that the minimum performance was 2K pps where the number of patterns was 64 when Packet

Table 3.3: Specification of PC-UNIXs used for evaluation

| | |
|---|---|
| CPU | Xeon 2.8 GHz |
| Memory | 2GB |
| HDD | 73GB |
| Bus | PCI-X (64bit, 133MHz) |
| Network Interface | 2 ports, 10/100Base-TX |
| Operating System | RedHat 9 Linux kernel 2.4.20 |

Buffer Process was deployed in the capture device. The appropriate number of patterns is expected to be between 4 and 64 in practice, therefore the performance per capture device would be between 2K to 15K pps. As described in Section 3.5.1, PCAP library was adopted for capturing packets. The timestamp is already obtained before the packet is buffered. The delay while packets are buffered doesn't cause inaccuracy on their arrival time.

On the other hand, the packet loss was observed on 500 pps traffic when the Packet Buffer Process was not deployed. From 2 experiments, the Packet Buffering Process is effective.

### 3.6.3 Performance Difference in Deploying AVL Tree Search or Sorted List Search

Figure 3.6 shows the capture device process performance when the range size of the bit-pattern is between 64 and 8192 respectively. The FID in packets which matched with the flow definitions was generated in a round-robin manner, so the number of each FID in packets became the same between FIDs. The length of the bit-pattern was four bytes.

From the results, the criterion to select a search from the AVL tree search or the sorted list search on the proposed system was 1024 in point of range size view. It may be effective if each flow definition statically selects one of searches such as the AVL tree search and the sorted list search with a criterion of the range size of the bit-pattern when the flow definition is registered in a capture device. The range size of a bit-pattern composing a flow definition

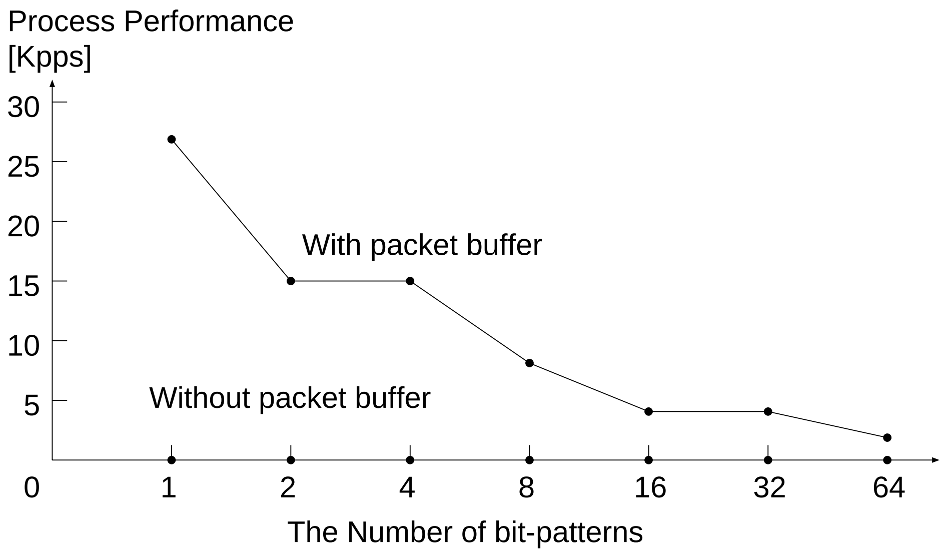


Figure 3.5: Process performance per number of bit-patterns with or without the packet buffer

is previously obvious when the flow definition is registered.

### 3.6.4 Scalability Evaluation

Figure 3.7 shows the average performance from 10 examinations for each number of bit-patterns and the capture devices. The exact pattern matching with 4-byte-length bit-patterns was used. All bit-patterns was specified at the same position in a packet. The FID field of generated traffic picked up by the capture device is generated in round-robin manner to match with the bit-patterns with the same frequency. Therefore every bit-pattern matched with 1 *n*-th of total traffic, where *n* is the number of given bit-patterns. The Flow Counter for the measurement attribute was used and the sorted list for the FID search to increment counters was adopted.

The performance increased in proportion to the number of capture devices and it reached 80K pps with six capture devices.

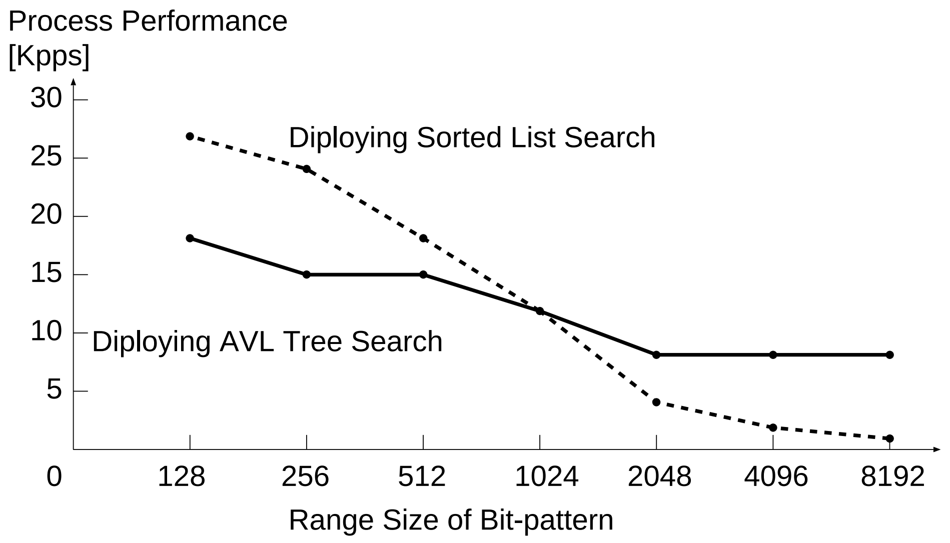


Figure 3.6: Process performance per range size of bit-pattern with AVL tree search or sorted list search

## 3.7 Discussions

### 3.7.1 Evaluation Results

From the results of Figure 3.5 in case the packet buffering process deployed, it is clear that the number of bit-patterns per packet significantly impacts on performance. The proposed system is assumed to perform with the multiple flow definitions composed of the multiple bit-patterns. The hierarchical bit-pattern structure is expected to prevent the performance degradation as the number of reduction of bit-patterns.

In order to improve the performance, it would be better to modify the order of chained bit-patterns operating as ’and’ operations between bit-patterns while the system is in progress. This tuning to reduce the average number of pattern matching per packet depends on the traffic measured. The dynamic matching order method is expected to reduce the entire pattern matching overheads.

An example of average packet length from a real network is showed for the reference of

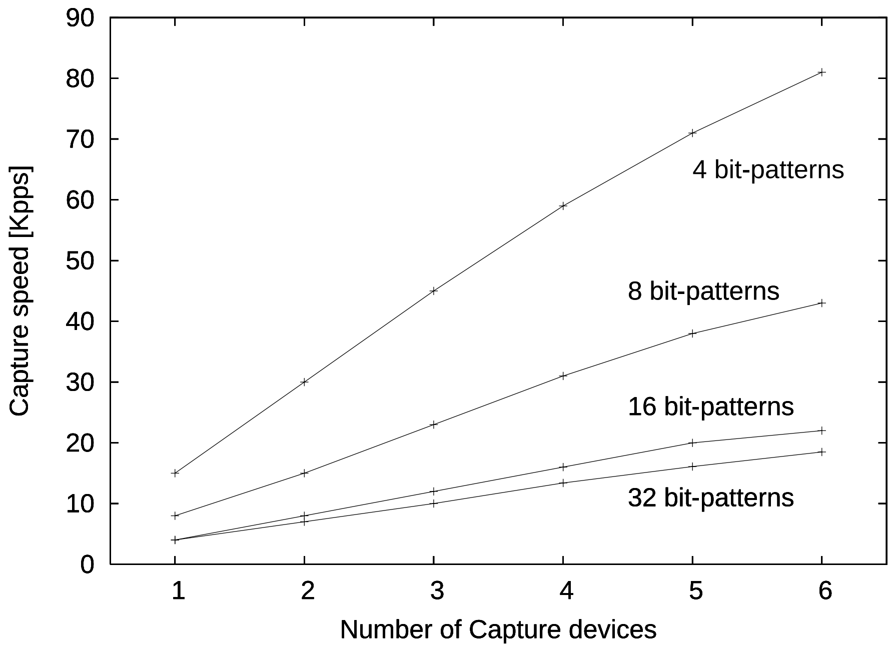


Figure 3.7: Performance of entire system

the proposed system performance. Figure 3.8 shows the 1 minute average packet length ('+') and average traffic ('×') of the traffic passing through a gigabit-ethernet link between APAN Tokyo XP [LW00] and WIDE [wid] measured in June 18, 2003. The average packet length on that day was 861 bytes. The results of this packet length are only an example, yet the performance of 2k – 15K pps per the capture device is estimated at 13M – 100M bit/s. 80K pps performed by 6 capture devices of Section 3.6.4 is estimated at 600M bit/s if observed traffic have 861 bytes which is same as the average packet size.

As showed in the Section 3.6.4, the performance of the entire system increases in proportion to the number of the capture devices and the proposed architecture addresses scalability by deploying multiple capture devices. However, the reason of a little performance falls at six capture devices is because once or twice of significant performance failure occurred in 10 examinations. The performance failure came from the rise of the delay of data processing at the threads of the Definition Advertisement and Collection Process in the manager device,

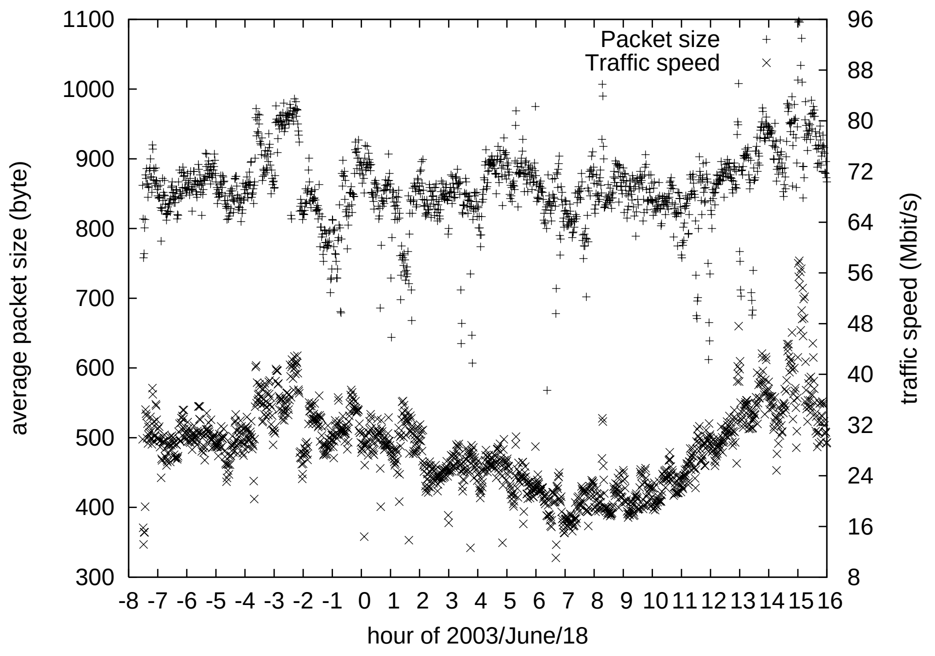


Figure 3.8: Packet length of traffic passing between APAN and WIDE

due to the large volume reports for Flow Counter from six capture devices. The long delay caused significant gap of data processing between capture devices and manager devices and thus resulted to discard delayed reports beyond a given threshold.

Using multiple capture devices in the proposed system is a key to achieving high-performance measurement of the IP flow on a high-speed link. However, the manager device can be a bottleneck for the performance of the entire system. The one of solution may be further to separate the components, the Status Preservation Component and the Data Preservation Component, to multiple devices.

Through all evaluations the proposed system showed that it provides scalability (Table 3.1) by achieving the requirements defined in Section 3.3.2. Improving the further performance scalability by investigating and developing the distribution device and the manager device as future works remains as future work.

### 3.7.2 Performance

As discussed in Section 3.3.2, existing hardware-based flow measurement system has limitations such as the number of flow definitions given at the same time to follow wire speed supported by its NICs. E.g. MD1230A series from Anritsu products accepts up to 4 flows [anrb]. On the other hand, the proposed system provide the interface to define flows flexibly and have no limitation the number of the flow definition and thus give users the convenience and flexibility in their operations. Additionally, although the proposed system composed by a few computers is inferior to the hardware-based system in the performance aspect, the proposed system can bring the performance close to hardware-based system by adding computers.

Comparing to existing software-based systems, Argus [BD], for instance measures flows specified by addresses from Datalink to Transport layers. Although Argus basically can measure all flows passing the observation point, it constraints users to obtain measurement data aggregated by addresses or port numbers such as IP prefix in the offline manner with proper scripts because it doesn't provide the interface for user to define flows used in the online manner. On the other hand, NeTraMet can collect measurement data aggregated by the combination of addresses and/or port numbers from Datalink to Transport layers due to the interface for user to define flows using these addresses. NeTraMet adopts hashing algorithms to handle flow identification process particularly for these address treatment to follow high traffic rate [Bro97].

As described in Section 3.4.3, besides the collection of aggregated flow data as NeTraMet provides, the proposed system provides better interface for the flow definition that user can flexibly specify flows using any field of packets. This flexibility enables us various operational applications such as to measure round trip time for TCP flows (described in Section 3.8.2, to monitor SYN flag of TCP to detect Syn Flood attack, to monitor connections carrying abnormal volume traffic and so on. To improve measurement performance in the condition supporting such flexibility in the flow definition, the data structure and architecture is designed for reducing the number of pattern matching per packet.

For compare these software-based systems including the proposed system, it’s hard to find the significance from the performance comparisons under the identical conditions because these systems have different design and implementation originating in their different usage fields. However, note that the performance of all systems rather depends hardware specification of computers on which the software of the systems implemented, at least. Under this point of view, the proposed system can follow the traffic rate without the dependency of the computers performance by adding the number of computer in the system, as the evaluation showed in Section 3.6.4.

### 3.7.3 Timestamp Accuracy and Distribution

The capture device employs its own clock in the implementation. As described in the Section 3.5.2, the timestamp becomes inaccurate in the software-based system. To measure high speed links, high time precision is required (e.g. at least 38 nanosecond to detect 48 byte frame on OC-192 link). Both delay variations of packet forwarding and software-based packet capturing are hardly accepted to meet such high speed links. Therefore timestamp process should be implemented in the distribution device of the hardware based implementation and both delay variation should be removed.

The timestamp function realized in a distribution device requires the capture device to obtain a timestamp from a packet and to shift the start or end point of the packet for the timestamp field. This also eliminates the need for clock synchronization between the capture devices and hence system becomes simpler.

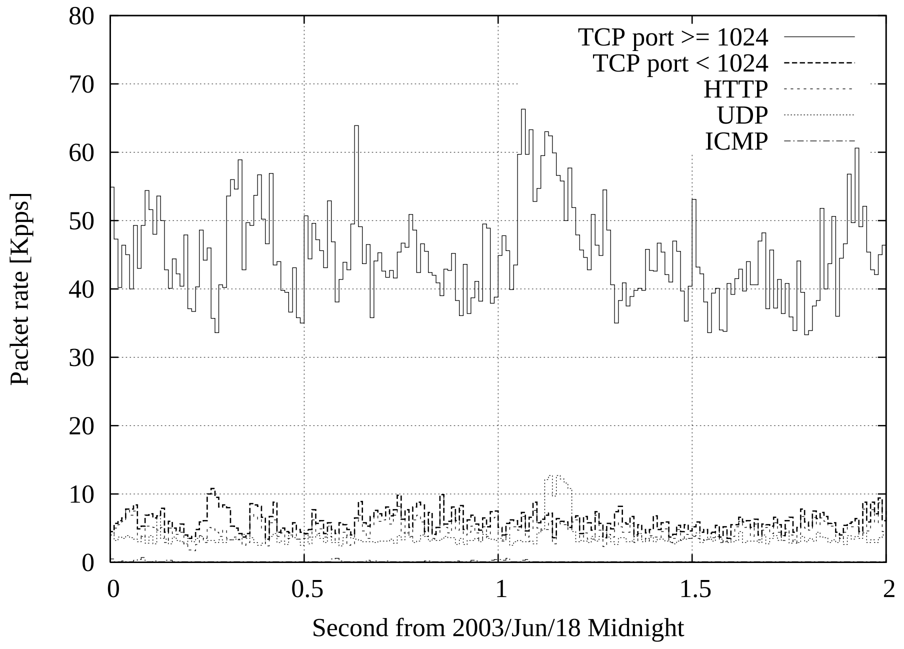


Figure 3.9: Flows of outgoing traffic to WIDE

## 3.8 Application and Advantage of Flow Measurement

### 3.8.1 Example Real World Environment

Figure 3.9 shows an example of the flow measurement of packet speed on the same gigabit-ethernet link between APAN Tokyo XP and WIDE as showed in Figure 3.8. The measurement was performed with a 2-capture-device system. The capture devices made reported every second. The prototype using PC-UNIXs distributed the input traffic in a round-robin manner. The total average and maximum traffic speed was 0.92K pps and 2.8K pps, respectively. The bit-pattern definitions were for checking the protocol field of the IP packet (1-byte length) and the destination port field of TCP header (2-byte length). There was no capturing loss detected during measurement. The resolution of the graph is in milliseconds.

The procedures for graph generation of Figure 3.9 with the proposed system were as follows:

1. Register flow definitions with the manager device
2. Measure flows with the multiple capture devices and store date to a manager device
3. Check graph for each flow definition by using the user interface device
4. Dump flow information in a specific period to text file
5. Generate a graph with an appropriate tool such as RRDTool [Oet], Gnuplot [WK], Microsoft Excel, etc.

Steps 1 to 3 were repeated until the graph that the user wanted was showed on the user interface device. It was easier to extract certain flows at the observation point because measurement based on the flow identification definition registered started just after its registration, and the graph showed up without delay. The operation with the proposed system effects to help operators look and feel for the traffic status without complex procedures.

### 3.8.2 RTT Measurement

RTT measurement is an example of the use of the Associated Packet Gap. RTT measured passively is not common because there is not always traffic passing a path of interest nor is it able to specify the probe packet size, interval of consecutive probes and measurement period. RTT itself is essential for the buffer size decision of high performance TCP transmission in a delay environment or for path selection for delay sensitive real-time applications such as VoIP or contents streaming. However RTT inference with an active measurement tool may cause competition between application traffic and the measurement traffic when measurement is performed before the application traffic is sent. The passive approach is more scalable and becomes more appropriate if the RTT inference for optimization of application traffic becomes more common [ZL03].

A TCP connection is suitable to measure RTT passively because a packet pair of data and its acknowledgment (ACK) can be the probe. To collect timestamps of the pair, the proposed

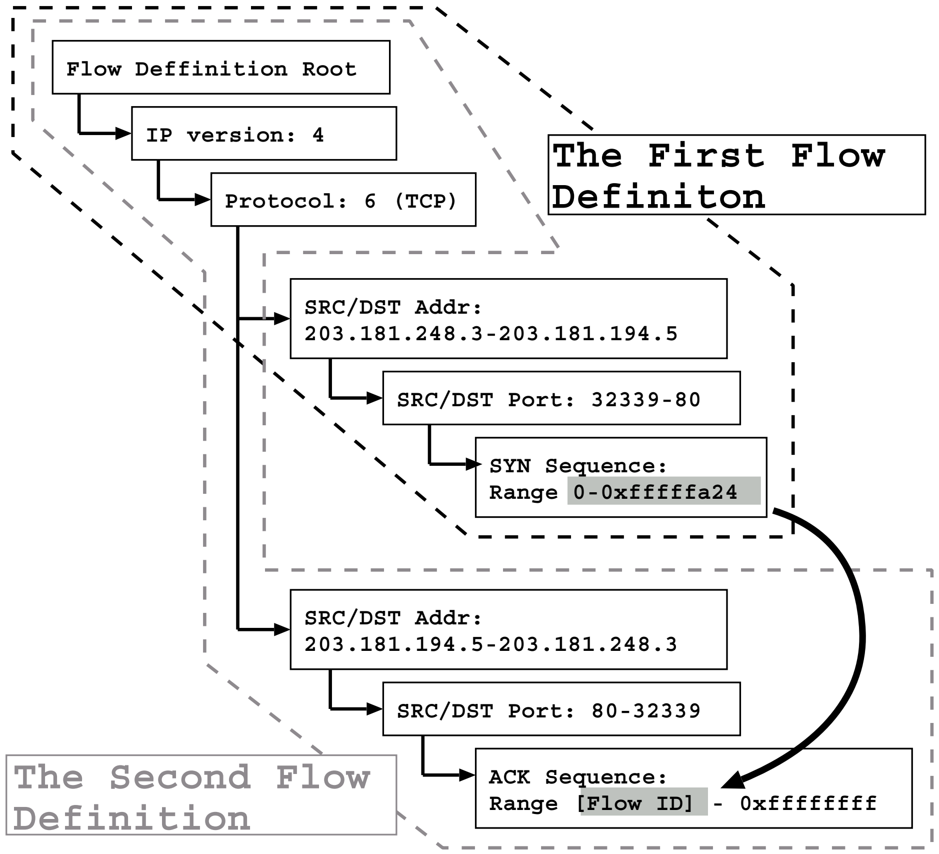

Flow Deffinition Root
IP version: 4
Protocol: 6 (TCP)
The First Flow Definiton
SRC/DST Addr: 203.181.248.3-203.181.194.5
SRC/DST Port: 32339-80
SYN Sequence: Range 0-0xfffffa24
SRC/DST Addr: 203.181.194.5-203.181.248.3
SRC/DST Port: 80-32339
The Second Flow Definition
ACK Sequence: Range [Flow ID] - 0xffffffff

Figure 3.10: An example of flow definitions to collect RTTs from a TCP connection

system needs to receive bidirectional traffic at an observation point and to have two flow definitions to specify data packets and its ACK packets, which have replaced source/destination addresses and ports between them. The first definition is used to collect the timestamp of a data packet and the FID matching with the sequence field bit-pattern. The second is for the timestamp used to compute interval between the data packet and itself matching with the ACK sequence field bit-pattern. Both definitions are connected by the Association Packet Gap attribute described in Section 3.4.4

Receiving the ACK sequence at a data source host implies that all the data sequences previously sent less than ACK sequence are received at a destination host, even though the

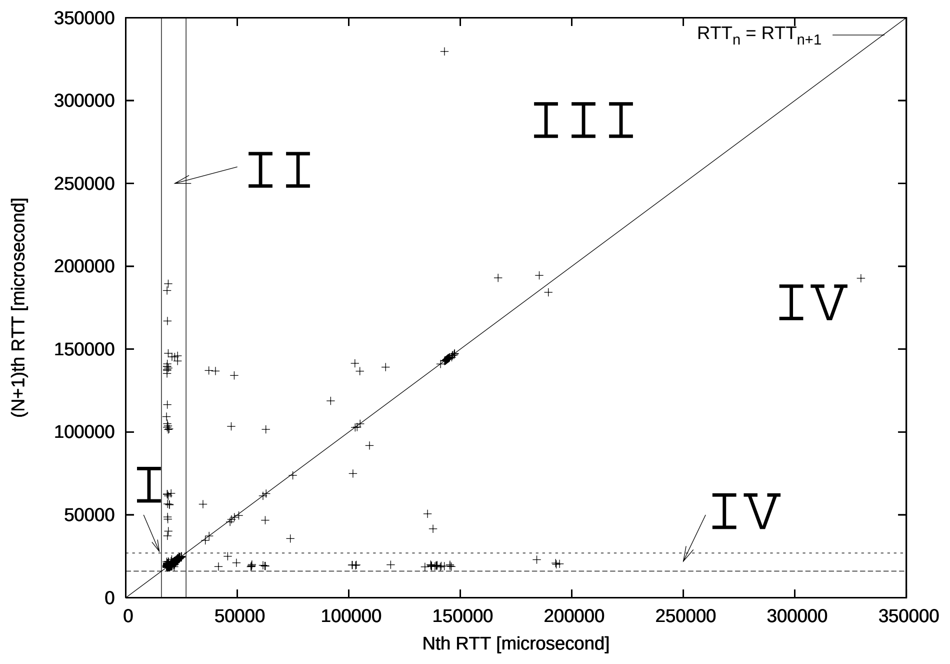


Figure 3.11: An Example of Phase Plot Showing Different Congestion Region

number of ACK sequences doesn't match with one of the data sequences exactly. Therefore the FIDs (sequence of data packet) collected by the first flow definition should be applied to the minimum value for range matching of the second flow definition to match with an ACK packet corresponding to the data packet. The maximum value of the second flow definition should be "$0xffffffff$." FIDs of the first flow definition around the highest portion of a sequence before rounding back 0 may not match the ACK sequence, because the highest ACK sequence portion may be omitted by the Delayed ACK [Cla82] (DACK) and round back to the low value. Therefore it is a good idea that the first flow definition uses a range matching not to collect the highest portion of the data sequence, such as the multiple size of MTU as an example. Figure 3.10 shows an example of two flow definitions bound by the Association Packet Gap attribute to measure RTT from a TCP flow.

Figure 3.11 shows an example of RTT phase plot [ZWJ03] showing minimum RTT realm, the correlation between two adjacent RTT values and congestion transition. The RTTs for this phase plot was collected by the proposed system from the traffic of both ways passing the same gigabit-ethernet link between APAN Tokyo XP and WIDE showed in Figure 3.8. The target flow lasted for 4.589 seconds and 1128 RTT data was collected. The X-axis ticks in that plot represent $RTT_n$ and the Y-axis ticks represent $RTT_{n+1}$, where $n$ and $n + 1$ are indexes of RTT data. Figure 3.11 shows that minimum RTT including TCP process time at end system is 8.3ms, minimum RTT realm ($E$) is 0.6ms. Zeitoun *et all*[ZWJ03] describes that three regions of $I$, $II$ and $III$ derived from the phase plot imply *no congestion, transient congestion* and *persistent congestion*, respectively, as follows:

- *Region I* contains probe pairs that experience minimum RTT and minor random overhead $E$ due to router, media or end host's TCP processing.
- *Region II* contains probe pairs that experience the beginning of congestion on the path.
- *Region III* contains probe pairs that experience persistent congestion due to packets are queued in routers or an end host raising process burden.

DACK and retransmission to use TCP's data and its ACK packets should be considered for the passive RTT measurement. DACK is done when a receiver acknowledges every multiple data packets received or when no acknowledgment is sent of more than 200 milliseconds. Therefore the region $II$ and line $R_n = R_{n+1}$ become filled by plots of enlarged RTT caused by DACK besides congestion on the path even though there is no loss with the target flow. The retransmission also shows large RTT in Region $II$ for the beginning of loss and $III$ for the rest of the following. For the accurate inference to detect congestion on the path, It is a good idea to extract minimum RTT from multiple RTTs computed from an ACK packet arrival. This extension requires small modification on the to the Association Packet Gap attribute, but a greater effect is expected.

## 3.9 Conclusions

The requirements for flow measurements having the following ability are discussed: helping network operators to understand how the network is being used in a spatially fine-grained view, e.g., not only total traffic volume but traffic volume per route or applications, and in a temporally fine-grained view, e.g., not only five-minute average but millisecond-order behavior, even on high-speed links. Based on them, an architecture of scalable and flexible real-time flow measurement tools was proposed. The architecture adopts multiple capture devices for processing packets in parallel, hierarchical chained flow definitions for reducing average overheads of pattern-matching, and an adaptive tree data structure for dealing with a large number of active flows.

To evaluate the proposed architecture, a proto-type implementation of the tool on PC-UNIX was developed, and examined its performance in testbed experiments, resulting in that the proto-type demonstrated to be able to measure statistics on flows defined by complex and on-demand definitions on a gigabit class link. Even by the proto-type system on nominal PC-UNIX machines, the maximum performance reached 80K pps with six capture devices.

The applications of measurement with the proto-type were demonstrated in real environments. The real-time fine-grained flow visualization enables ones to understand what types of traffic are dominant and, at the same time, how they are behaving over a short time-scale. Monitoring bidirectional TCP flows enables ones to infer RTT behavior without sending any probe packet.

The development of the distribution device and the evaluations of its performance and that of entire system require further investigation.

# Chapter 4

# Influence of network characteristics on application performance in a Grid environment

## 4.1 Introduction

With the availability of high-performance off-the-shelf computers and high-speed wide-area networks, large-scale distributed computing environments are growing at an amazing speed. These environments enable massive computations to be performed using a large number of computers connected over WAN (Wide Area Networks). This form of distributed computing (grid computing) dynamically involves a number of heterogeneous computing resources (e.g., CPU, memory, storage, application program, data, etc.) connected by heterogeneous network resources across geographically dispersed organizations [FK98] [FKT01]. The fundamental challenge for grid computing is to achieve a good performance from multiple distributed applications that share limited and/or heterogeneous computing and network resources (e.g., in terms of the completion time for each separate application and/or set of related applications). Large amounts of data are now being handled by distributed applications and the time required to transmit data is increasing, which has a significant effect on the performance of an application. Effective share of network resources is therefore critical.

There is considerable work on traffic engineering to improve the utilization of network resources. However, there is little insight on to improve the application performance. The literature [EJLW01] proposed a decentralized method to balance flows over multiple paths based on the traffic load of the paths obtained by active end-to-end measurement along the paths. The literature [GKL$^+$04] proposed that routers re-balance flows over multiple paths with the information collected by a measurement device which passively measured loss and available bandwidth of connected links. Although both suggested re-balancing traffic over multiple paths, they neither considered the traffic patterns generated by various applications nor the application-level performance.

As focusing on improving the application performance, the literature [Kaw04] proposed equalizing throughput of TCP connections established in a path in terms of improvement of the total performance of data transmitted, e.g., by reducing the time of file transmission. However, the requirements of the network resources allocated to applications are diverse. Some applications may require a broadband path to perform effectively, and others may a small delay one instead.

Another direction of related work focuses on improving the application-level performance. The literature [Rao01] proposed a method based on estimating delay regression to achieve low end-to-end delays for message transmissions by using two paths for distributed computing applications. This method does not allow for the fact that the application-level performance is influenced by both the delay and available bandwidth of the selected path and may, thus, unnecessarily assign an application a short-delay path from which sufficient bandwidth can not be obtained.

The literature [AO05] illustrated the case study on improving the performance of a master-worker application, which was grouped into a work flow process (described in Section 4.2), by localizing the communication frequently occurred, into a PC cluster or a single PC so as to reduce the communication influenced by a long delay. Plaat *et all*[APH99] investigated the impact of a limited bandwidth and a long delay on the application-level performance for 6 applications, and suggested how to optimize the algorithms of them to weaken the sensitivity of application performance to a limited bandwidth and a latency, and, thus, to improve their performance under the limited network resource conditions. They mostly focused on optimizing the task allocation of target applications to multiple computers to reduce the effect of state of network resources on the distributed computing applications. However, in case that multiple applications simultaneously run in a dedicated network, they neither consider influence of the traffic generated by individual applications on the other applications nor how to share the limited network resources among them to obtain the better performance in total.

Suppose that the network-related properties of various applications could be determined in advance based on either a test run or the first run in a series of repeated runs. There are two aspects in the network-related properties of an application i.e., how the performance of individual applications is affected by the state of the network resources and, conversely, how traffic generated by individual applications affects the state of the network resources. A scenario in which the network-related properties of applications are taken into account may allow these distributed applications to be scheduled to share the network resources on an internal-network more effectively as well as the computing resources at the end-nodes. To address this issue, therefore, applications are investigated if some of them show specific performance characteristics in relation to the network. The sensitivity of some typical distributed applications is focused on in terms of completion time under different network resource conditions, and on the impacts of traffic generated by distributed applications on other traffic sharing the same bottleneck and on the performance of other applications. The results show a non-trivial tension between the performance of an application and the network resource conditions, which can be utilized to achieve more effective sharing of the network resources, allowing multiple applications to be executed in parallel.

## 4.2 Distributed Computing Applications

In this chapter, four distributed computing applications as follows are employed. Each application displays the different traffic features.

- *N Queen* solves the placement of N Queens on an N by N grid such that none of them shares a common row, column or diagonal.
- *Jigsaw Puzzle* solves a jigsaw puzzle for computers, which comes originally from Problem C in the 2001 ACM International College Programming Contest, Asia Preliminaries in Hakodate [icp01] and which has been expanded to take the rotation and size of pieces into account.
- *LU Decomposition* is one of the programs in the NAS Parallel Benchmarks [BBB+91] for solving a lower triangular matrix, **L**, and an upper triangular matrix, **U**, composing an $N \times N$ matrix **A** (= **LU**).
- *Task Scheduling* is a task scheduling program that deals with standard task graph archives [TK02]; it assigns a task to a computer that has sufficient available memory based on the priority given to a task belonging to a task flow that takes longer to be processed.

N Queen, Jigsaw Puzzle, and LU Decomposition are classified as task-framing programs in terms of type of distributed computing and Task Scheduling is classified as a work flow program [Buy99]. In the task-framing process, a master distributes the tasks that make up the target problem among a farm of multiple slave processors and gathers the partial results to produce the final result of the computation. Generally, communication between the master and slaves involves huge transfers of data. In work flow processing, the target problem is divided up into multiple pipelined stages and/or steps, and various amounts of data are asynchronously exchanged between pairs of processors.

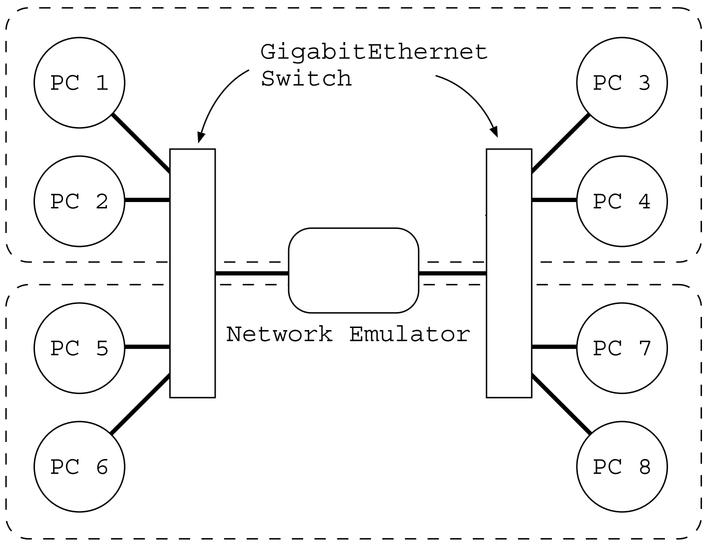


Figure 4.1: Network configuration

## 4.3 Experimental Environment

The network configuration showed in Figure 4.1 are employed for the experiments described in this chapter. The distributed processing for each application described in Section 4.2 was performed on computers PC1 through PC4 starting from PC1, or on PC5 through PC8 starting from PC5. All the computers had the following basic specifications: Xeon 3.06 GHz CPU, 2 GBytes memory, Intel (R) PRO/1000 NIC and PCI-X bus. The speed of all links was 1 Gbit/s.

For the task-framing applications, the master process ran on PC1, and the slave processes ran on all the computers including PC1. A network emulator [pac] was used to insert latency and to shape a bottleneck link in packet forwarding between two switches. Measurements were performed by capturing all the packets sent to or received from each of the computers. The average round-trip times (RTTs) were 0.141 ms and 0.331 ms between PC1 and PC2, and between PC1 and PC3, respectively, without any latency inserted.

Table 4.1: Feature of traffic to/from a computer for each application

| Application | Incoming traffic to PC3 | | Outgoing traffic from PC3 | | Completion time |
|---|---|---|---|---|---|
| | Total amount of data | Average throughput | Total amount of data | Average throughput | |
| N Queen 15×15 grids | 0.926 MB | 0.878 Mbps | 39.2 MB | 37.2 Mbps | 40.78 s |
| N Queen 16×16 grids | 4.00 MB | 0.167 Mbps | 242.05 MB | 10.12 Mbps | 191.33 s |
| Jigsaw Puzzle 4×4 pieces | 0.0780 MB | 0.0443 Mbps | 0.117 MB | 0.0666 Mbps | 13.70 s |
| Jigsaw Puzzle 36×36 pieces | 101 MB | 4.56 Mbps | 194 MB | 8.79 Mbps | 176.48 s |
| LU Decomp. 64×64 mtx | 67.45 MB | 10.37 Mbps | 67.21 MB | 10.31 Mbps | 50.18 s |
| LU Decomp. 102×102 mtx | 179.07 MB | 5.24 Mbps | 178.90 MB | 5.23 Mbps | 258.64 s |
| Task Sched. 300 tasks | 116 MB | 7.68 Mbps | 132 MB | 8.78 Mbps | 131.83 s |
| Task Sched. 500 tasks | 163 MB | 10.1 Mbps | 188 MB | 11.6 Mbps | 132.52 s |

## 4.4 Analysis of Communication Features

The features of the application traffic were investigated, e.g., the amount of transferred data, fluctuations in throughput and the flow composing the application traffic.

Table 4.1 show the total amount of transmitted data, average throughput, and completion time for each of the applications. The transmitted data and throughput were measured at PC3 when an application was run on PC1 through PC4 with the sufficient network resources, the average RTTs were 0.114 and 0.331 ms and the bandwidth of links was 1 Gbit/s. The values in the table are the average of 20 experiments.

All applications take longer to complete their processing and transmit larger amounts of data as the scale of the problems becomes larger. The relationship between the amount of

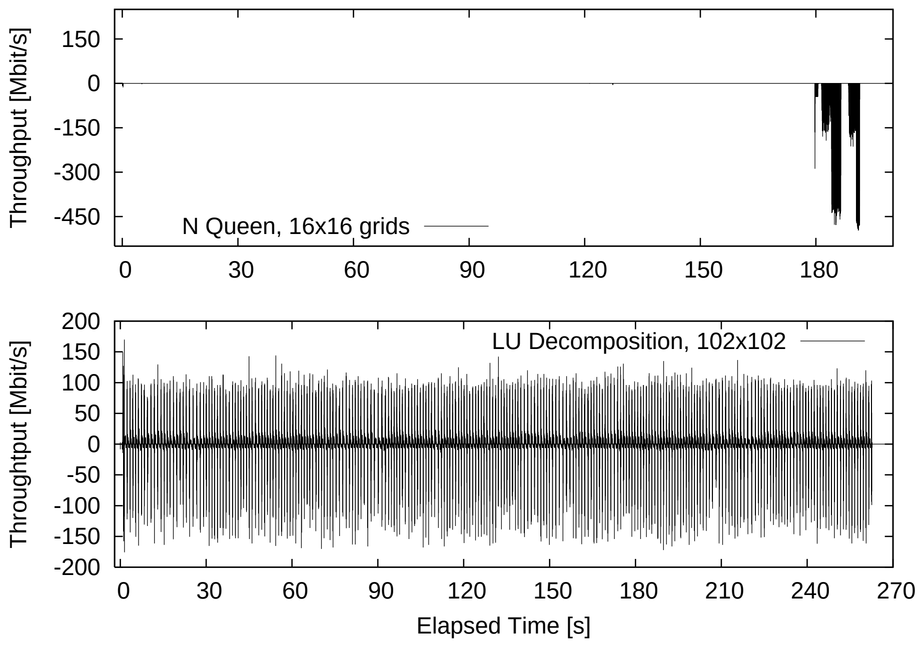


Figure 4.2: Fluctuation of throughput of NQueen and LU Decomposition. X-axis is the elapsed time in seconds after applications start. Throughput of 10 ms average is on y-axis. Positive values on y-axis indicate traffic incoming to PC3 while negative values indicate outgoing traffic.

data transmitted and the completion time greatly depends on the application. For N Queen, increasing the amount of data 6 times increases the completion time 6 times. Jigsaw Puzzle shows a 10-fold increase in completion time when the data increases significantly by more than 1000 times. The completion time for Task Scheduling scarcely changes while the amount of data is increased by 1.5 times. For LU Decomposition, in contrast to the other applications, the average throughput decreases by half while the completion time and amount of transmitted data are increased 5 and 3 times, respectively. The completion time for Jigsaw Puzzle and Task Scheduling are less affected by an increased amount of data, probably because the average throughput increases to convey larger amounts of data.

Note that in the following experiments, the results of using the 16$\times$16 grids for N Queen,

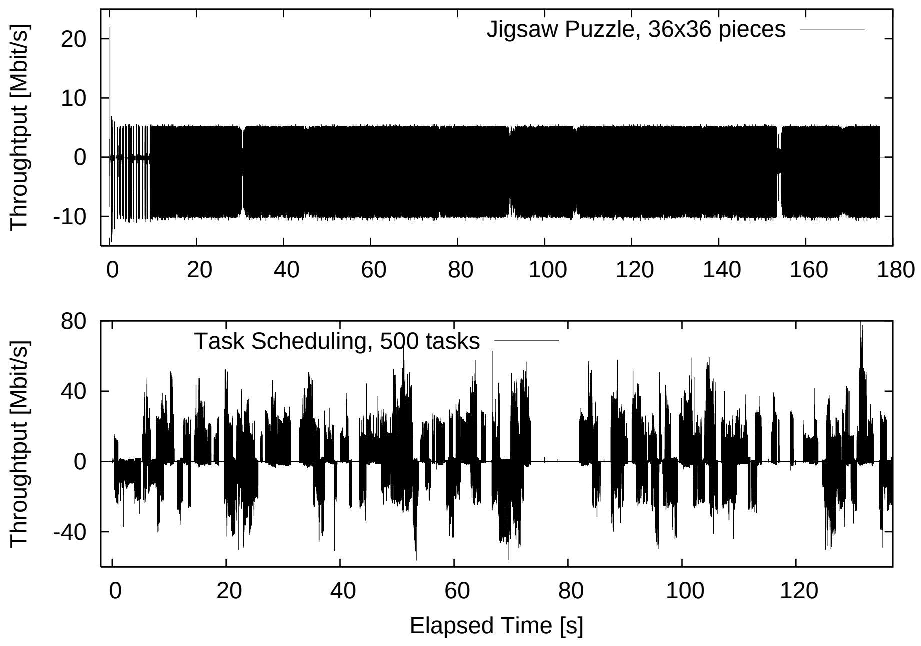


Figure 4.3: Fluctuation of throughput of Jigsaw Puzzle and Task Scheduling. X-axis is the elapsed time in seconds after applications start. Throughput of 10 ms average is on y-axis. Positive values on y-axis indicate traffic incoming to PC3 while negative values indicate outgoing traffic.

36×36 pieces for Jigsaw Puzzle, 102×102 elements for LU Decomposition, and 500 tasks for Task Scheduling are presented because the alternative cases tended to produce quite similar results to these.

Figures 4.2 and 4.3 show the fluctuations in the 10-ms-average throughput of data transmitted to/from PC3 for one instance of the 20 experiments described in Table 4.1. Note that both PC2 and PC4 show a similar traffic pattern to PC3 for all the applications. However, the patterns between applications were completely different among applications. N Queen sends a huge amount of data from the slave to the master (outgoing from PC3) near the end of the process. Jigsaw Puzzle continuously exchanges data at a stable rate, 5 and 10 Mbit/s throughout processing. LU Decomposition also exchanges data continuously, but its

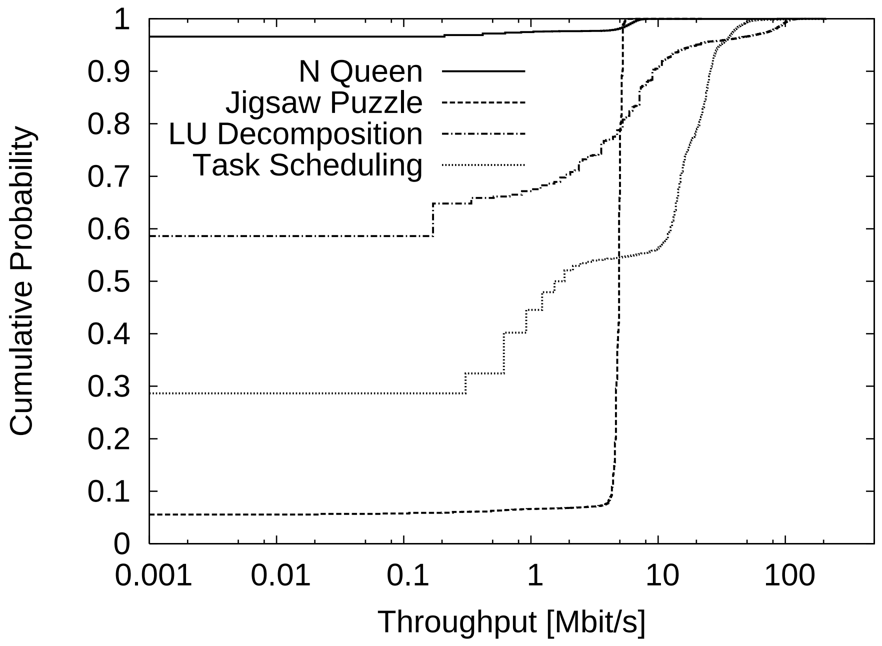


Figure 4.4: Cumulative probability of throughput of input traffic for each application

throughput behaves in an on-off manner, and varies within a short period. Task Scheduling intermittently exchanges large amounts of data throughout processing.

Figures 4.4 and 4.5 show the cumulative probability of the throughput of incoming and outgoing traffic, respectively, for each application for one instance of the 20 experiments.

For each application, the distributions of incoming and outgoing traffic are similar. For N Queen, more than 95% of the processing time is no traffic or less than 1 Kbit/s, while the second most frequent rate of throughput is near maximum. For Jigsaw Puzzle, the peak values are about 5 Mbit/s for incoming traffic and about 10 Mbit/s for outgoing. For LU Decomposition, for both incoming and outgoing traffic, about half of the throughput is less than 1 Kbit/s and the rest varies. Task Scheduling generates a variety of levels of throughput, with more than 40% of it being between 20 and 50 Mbit/s.

The feature of flows consisting of traffic generated by each of the applications. was

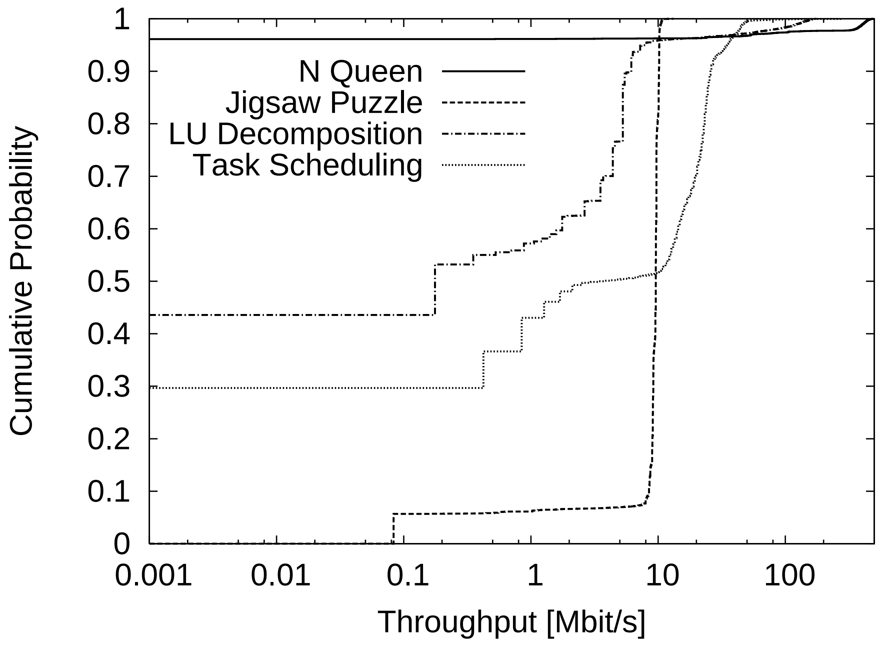


Figure 4.5: Cumulative probability of throughput of output traffic for each application

analyzed. All the applications used only TCP for task communication. Therefore, a flow is defined as a set of packets transmitted via a TCP connection by direction, beginning with a SYN flag and terminating with a FIN flag. Multiple TCP connections were often established in parallel through the processing in each application.

Figure 4.6 and 4.7 show the amount of transferred data and duration of each of the flows for N Queen and Task Scheduling for one instance of the 20 experiments, respectively. Jigsaw Puzzle and LU Decomposition show features similar to those of N Queen, as described in Figure 4.6, and which presumably relates to the fact that those applications are all categorized as task-framing types.

There are a much smaller number of flows for N Queen than Task Scheduling. Some flows last for less than 10 ms, some for about 10 s, and others from start to finish. The long-lived flows transmit various amounts of data from 100 bytes to more than 100 Mbytes. The

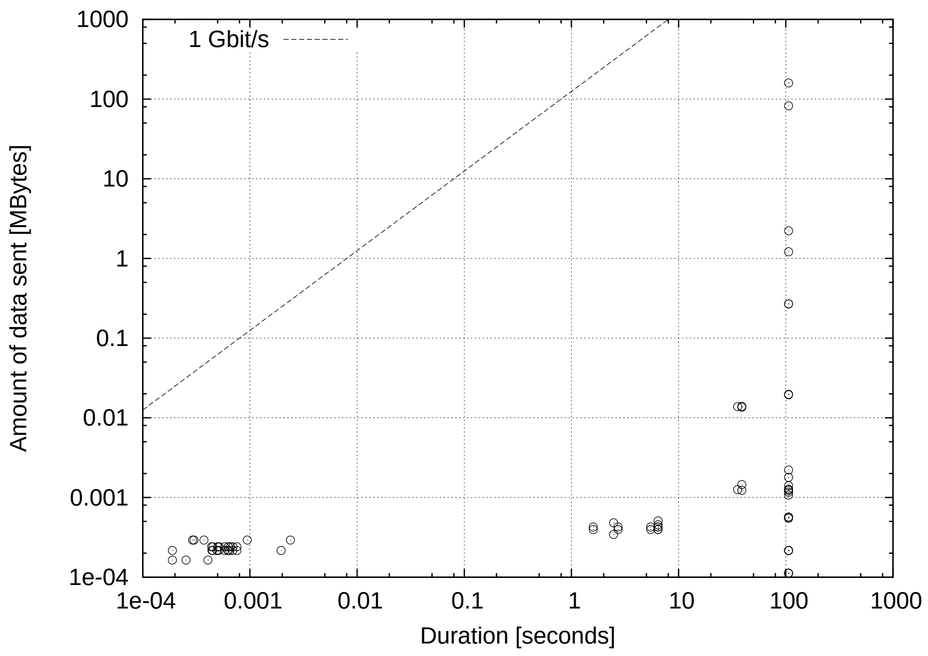


Figure 4.6: Duration and amount of data transferred in each flow generated in N Queen: Line is boundary of plots and is equivalent to 1 Gbit/s.

huge amount of data transferred by N Queen must be carried by these long-lived flows. As showed in Figure 4.7, there is a large number of flows varying duration for Task Scheduling, which transmits various amounts of data.

In the following sections, first, the influence of the network properties on application-level performance, is examined. Secondly, the impacts of the traffic from each of the four applications on other traffic sharing the bottleneck link are examined.

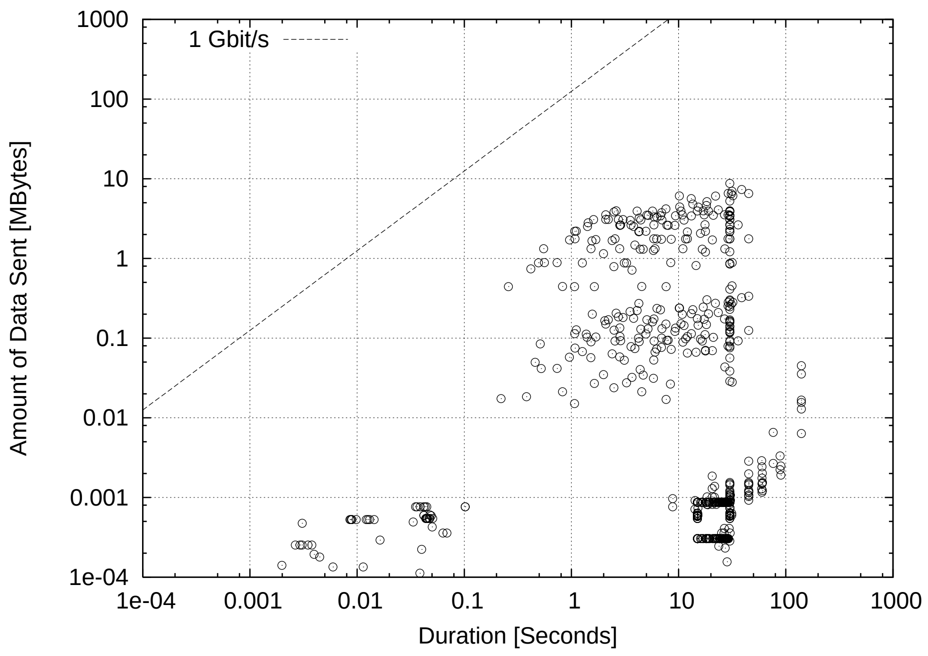


Figure 4.7: Duration and amount of data transferred in each flow generated in Task Scheduling: Line is boundary of plots and is equivalent to 1 Gbit/s.

## 4.5 Influence of Network Properties on Application-level Performance

To clarify the effect of the network properties on the application-level performance, the characteristics of the completion time for each application by imposing either a long RTT, a bottleneck link with a narrow bandwidth, or both on the application traffic were examined.

### 4.5.1 Impact of Large Round-trip Time

The influence of a long RTT on the completion time for each application running on PC1 through PC4, as described in Figure 4.1, was investigated. In the experiments, 1 to 32 ms latencies were inserted into traffic passing both ways using the network emulator showed in

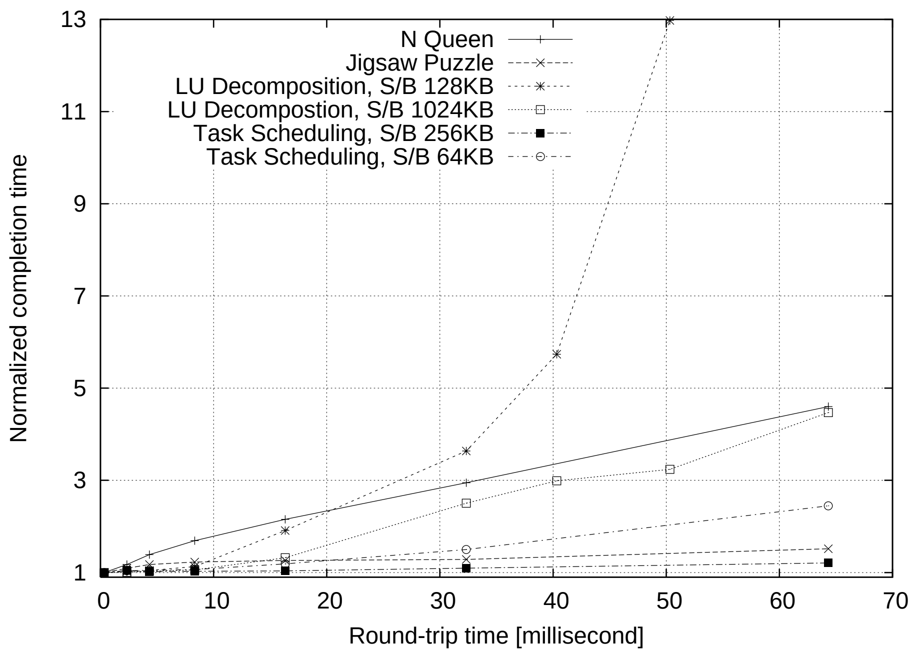


Figure 4.8: Completion time influenced by RTT: Application-level performance deteriorates as RTT increases.

Figure 4.1. The link bandwidth was configured to a value of 1 Gbit/s so that the focus was on the impact of the RTT on the performance of the applications. TCP socket buffers of 16 and 128KB in length were used in N Queen and Jigsaw Puzzle, 128 and 1024KB in LU Decomposition, and 64 and 256KB in Task Scheduling. Figure 4.8 shows the completion time for each application influenced by the RTT. The completion times are normalized by that obtained when these is no latency. Each completion time is the average of the results of 20 experiments.

The characteristics of the completion time for both N Queen and Jigsaw Puzzle were almost the same in the case of both the 16 and 128KB TCP socket buffers. The results for using a 128KB socket buffer are therefore omitted from Figure 4.8.

For all the applications, the application-level performance deteriorates as the RTT increases. For LU Decomposition and Task Scheduling, the large socket buffer is very effective in reducing the completion time, probably because the average size of the TCP sending window is equal to the product of the average throughput and the RTT in successive sending of large amounts of data. In the following experiments, therefore, 1024KB and 256KB socket buffer were used for LU Decomposition and Task Scheduling, respectively. N Queen and LU Decomposition are still influenced by the RTTs despite the use of a large socket buffer. And Jigsaw Puzzle is scarcely influenced by the RTTs even though it does not use large socket buffers. To try to determine the cause of these differences in the influence of the RTT on N Queen, Jigsaw Puzzle, and LU Decomposition, the interval between consecutive packets and the total amounts of data transmitted was analyzed

Figures 4.9 and 4.10 show the distribution of intervals between consecutive packets in flows generated between PC1 and PC3 for N Queen, and Jigsaw Puzzle, respectively. The results for LU Decomposition are omitted because LU Decomposition showed the change of the distribution of packet intervals in between the cases of small and large RTTs, similar to that of N Queen.

For N Queen (Figure 4.9), the distribution of packet intervals in the case of a short RTT (0.331 ms) shows that almost all the intervals are less than the RTT. This implies that almost all the packets are sent in succession. The packet intervals show two peak values for, one at about 0.012 ms corresponding to back-to-back data packets of 1500 bytes (more than 70% of intervals), and the other at about 0.1 ms. In the case of a large RTT (32.3 ms), the number of intervals in the most burst case (i.e., back-to-back packets) decreases to 60% and the intervals near the RTT increase instead. This shift indicates that a long RTT prevents rapid expansion of the sending window, resulting in a decrease in the throughput.

For Jigsaw Puzzle (Figure 4.10), the distribution of intervals also shows two peak values. One, which accounts for more than 40%, is roughly close to the interval between back-to-back packets of 1500 bytes (0.012 ms), and the other is roughly close to the RTT (0.331 ms)

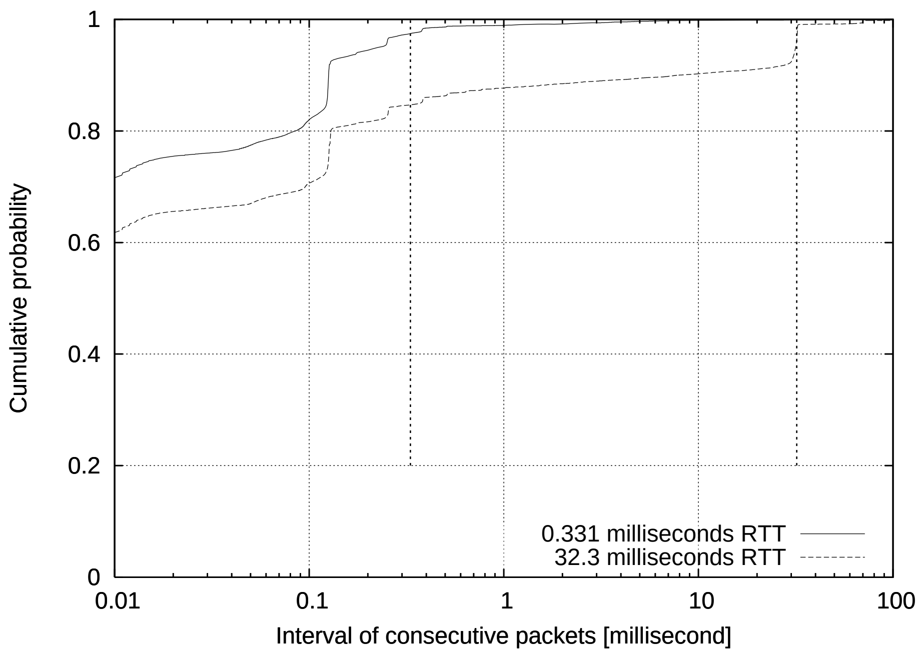


Figure 4.9: Distribution of inter-packet gaps in flows generated by N Queen

for sending packets interactively. In the case of a 32.3-ms RTT, the data is still transmitted in a burst manner, while the peak value corresponding to an RTT of 0.331 ms moves to a new RTT of 32.3 ms. The reason that Jigsaw Puzzle does not take so long to complete its processing when the RTT increases although this interactive communication is influenced by a large RTT may be because the total amount of transmitted data decreases as the RTT increases, as showed in Table 4.2.

The information on the sensitivity of application performance to a long RTT will be useful in selecting the appropriate path for different applications in multi-path environments.

### 4.5.2 Impact of Limiting the Bandwidth of the Bottleneck Link

the influence of limiting the bandwidth of the bottleneck link on the completion time of each application running on PC1 through PC4, as described in Figure 4.1, was investigated. In

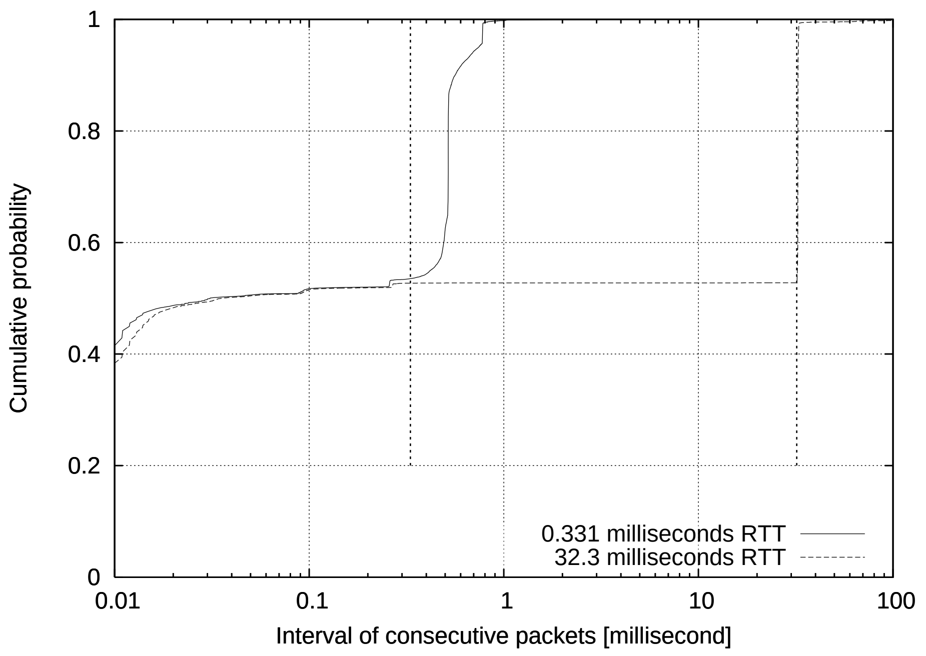


Figure 4.10: Distribution of inter-packet gaps in flows generated by Jigsaw Puzzle

the experiments, the bandwidth of the bottleneck link was varied from 80 Kbit/s to 1 Gbit/s using the network emulator in Figure 4.1. The RTT was the original short value without any additional latency so that the focus was on the impact of bandwidth restriction on the application performance.

Figure 4.11 shows that even if the bandwidth is reduced before reaching a threshold (a gradual deterioration threshold) for each application, the completion time remains roughly the same (less than 1.1). Furthermore, the completion time increases abruptly, exceeding 1.50 if the bandwidth is reduced after reaching a threshold (a severe deterioration threshold) for each application. Table 4.3 shows both values for each of the applications.

The information on the thresholds for limiting bandwidths will be useful in determining how limited amounts of network bandwidths should be allocated to various applications.

Table 4.2: Number of packets transferred in applications influenced by various round-trip times

| Application | RTT: 0.3 ms | | RTT: 4.3 ms | | RTT: 16.3 ms | | RTT: 64.3 ms | |
|---|---|---|---|---|---|---|---|---|
| | pkt | byte | pkt | byte | pkt | byte | pkt | byte |
| N Queen | 233K | 242M | 234K | 242M | 227K | 238M | 225K | 237M |
| Jigsaw Pzl | 2372K | 297M | 992K | 118M | 293K | 35.1M | 84K | 9.99M |
| LU Decomp. | 674K | 475M | 716K | 513M | 739K | 537M | 748K | 542M |
| Task Schd | 823K | 338M | 898K | 362M | 720K | 313M | 690K | 352M |

Table 4.3: Bottleneck bandwidths at which completion time deteriorates gradually or severely

| Application | Gradual deterioration | Severe deterioration |
|---|---|---|
| N Queen | 140 – 500 Mbit/s | < 40 Mbit/s |
| Jigsaw Puzzle | 30 – 200 Mbit/s | < 7 Mbit/s |
| LU Decomposition | 140 – 200 Mbit/s | < 70 Mbit/s |
| Task Scheduling | 40 – 200 Mbit/s | < 25 Mbit/s |

### 4.5.3 Impact of both Expanding RTT and Limiting Bandwidth of Bottleneck Link

the influence of both limiting the bandwidth of the bottleneck link and a large RTT on the completion time of each application running on PC1 through PC4, as described in Figure 4.1, was investigated. In the experiments, the bandwidth of the bottleneck link was varied from 10 Mbit/s to 1 Gbit/s and the latency from 1 to 32 ms using the network emulator in Figure 4.1. Figures 4.12 – 4.15 show the completion times for each application under the condition of a large RTT and a narrow bandwidth.

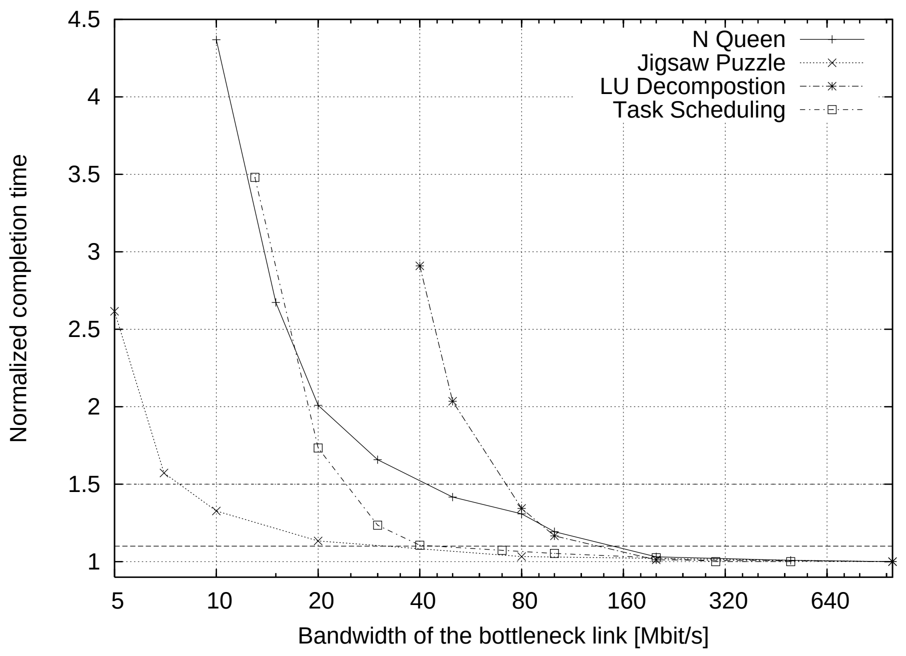


Figure 4.11: Completion time influenced by narrow bottleneck link

The deterioration of the completion time influenced by both a long RTT and limiting the bandwidth differs depending on the applications. For all the applications, the completion time deteriorates as the RTT increases even though the bandwidth of bottleneck link is sufficient, as described in Section 4.5.1. N Queen and Jigsaw Puzzle show that the completion time shifts higher as the RTT increases, and the change of the completion time, as the bottleneck link become narrower, shows similar tendency. The gradual deterioration threshold of the bandwidth in each of the RTTs, (at which the completion time deteriorates gradually) shifts lower as the RTT becomes larger, resulting in 30 Mbit/s in the case of a 64.3-ms RTT for N Queen, and in 10 Mbit/s for Jigsaw Puzzle. Note that, however, for LU Decomposition and Task Scheduling, the gradual deterioration threshold shifts higher as the RTT increases, resulting in 200 Mbit/s in the case of a 64.3-ms RTT for LU Decomposition, and in 180 Mbit/s for Task Scheduling. For LU Decomposition, the condition of a small RTT and a

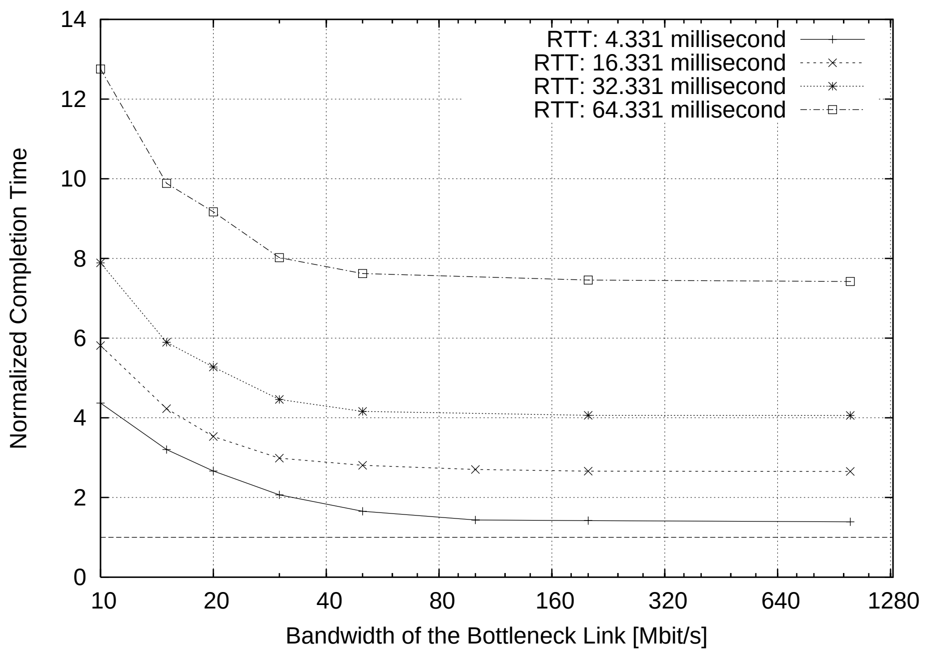


Figure 4.12: Completion time influenced by the narrow bottleneck link and RTT for N Queen

narrow bandwidth of the bottleneck link complicatedly influenced on its process and, as a result, the completion time in the case of a 4.3-ms RTT exceeded that of a 16.3-ms RTT when the bandwidth of the bottleneck link was 10 Mbit/s.

The information on the extent of deterioration and the shift in the thresholds for limiting the bandwidth will be useful in determining which path to select from multiple-paths with different RTTs and/or bottlenecks with different bandwidths.

## 4.6 Influence of Application Traffic on Other Traffic

the influence of application traffic on other traffic, that shared a bottleneck link with the application, was investigated. First the influence on packet loss ratio for UDP traffic was

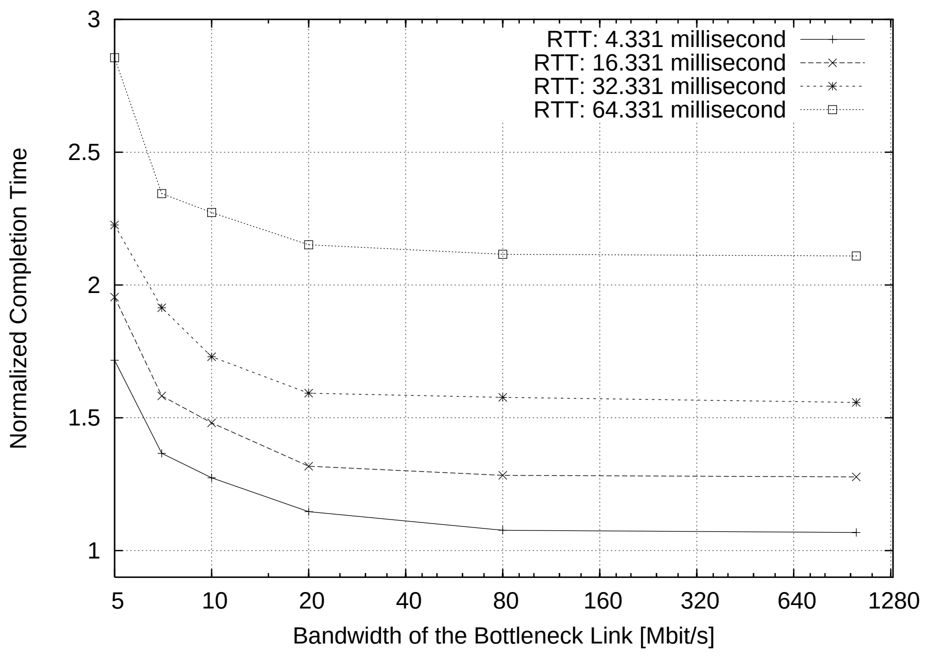


Figure 4.13: Completion time influenced by narrow bottleneck link and RTT for Jigsaw Puzzle

examined. Secondly the influence on application-level performance of executing two applications simultaneously was examined.

## 4.6.1 Influence of Application Traffic on UDP Traffic

To investigate the influence of application traffic on the state of network resources, the packet loss for UDP cross traffic, while one of the applications was running, was assessed. In the experiments, the cross traffic was generated by iperf [ipe] from PC7 to PC5, or from PC6 to PC8 while the application ran on PC1 through PC4, as described in Figure 4.1. The bandwidth of the bottleneck link was limited to 200 Mbit/s and the RTT was the original short value without any additional latency. Figure 4.16 shows the average packet-loss ratio of cross traffic at each bit rate from 100 to 190 Mbit/s. The top graph shows the results of

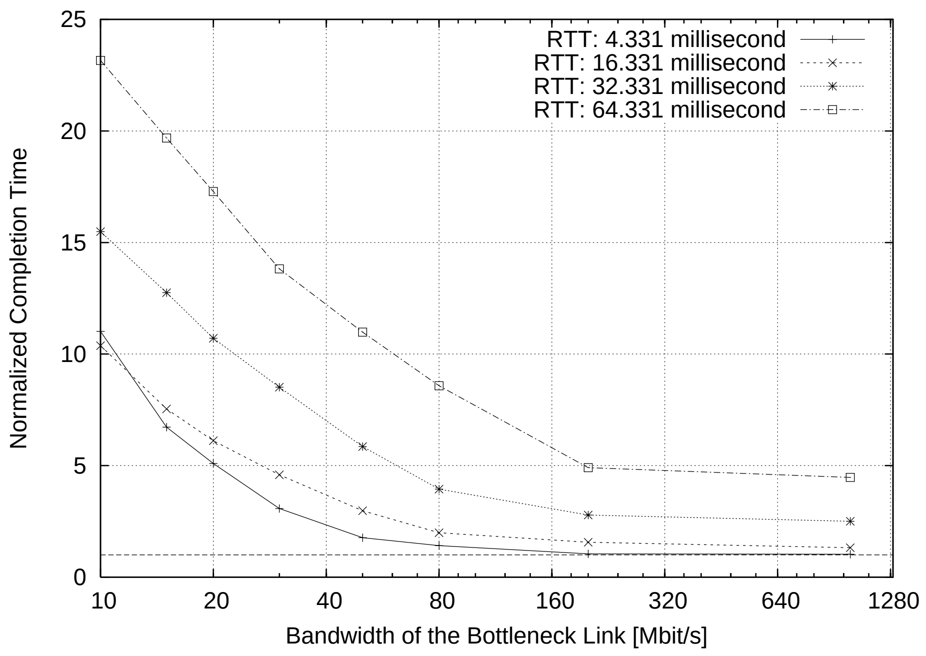


Figure 4.14: Completion time influenced by narrow bottleneck Link and RTT for LU Decomposition

cross traffic from PC7 to PC5 and the bottom graph from PC6 to PC8. These results are the average of 20 experiments.

The packet-loss ratio differs depending on the application, and the direction and rate of the traffic that it generates. The traffic patterns are given in Figures 4.2 and 4.3. High-rate cross traffic sharing the bottleneck link with N Queen traffic or Task Scheduling experiences significant packet loss. In particular, N Queen experiences severe packet loss in the direction from PC7 to PC5, which is the same as the large amount of traffic sent from slaves to master, as described in Figure 4.2. Cross traffic sent simultaneously with LU Decomposition also experiences significant packet loss when the rate is more than 140 Mbit/s both ways. Cross traffic sent simultaneously with Jigsaw Puzzle experiences much less packet loss both ways even though the rate increases.

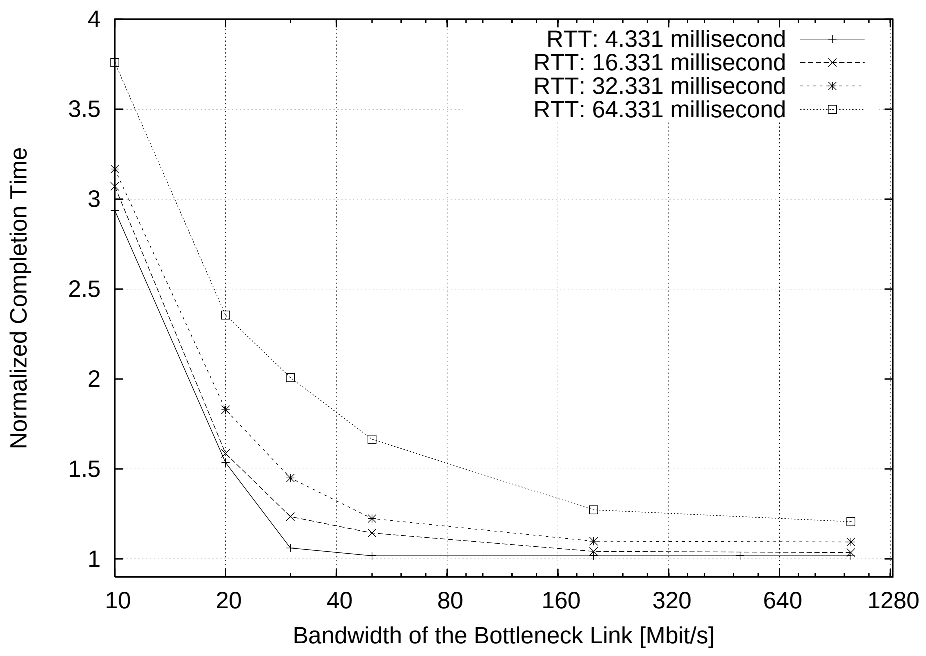


Figure 4.15: Completion time influenced by narrow bottleneck link and RTT for Task Scheduling

The information on the influence of traffic generated by various applications on other traffic will be useful in considering how to prevent adverse affects on traffic transmitted simultaneously with application traffic.

### 4.6.2 Influence of Application Traffic on Performance of Other Applications

The effects of executing two applications simultaneously on the application-level performance, were examined. In the experiments, one of the applications ran on PC1 through PC4 and the other on PC5 through PC8. The bandwidth of the bottleneck link was varied from 10 Mbit/s to 1Gbit/s using the network emulator in Figure 4.1, while the original short value (0.331 ms and 0.114 ms) was used for the RTT.

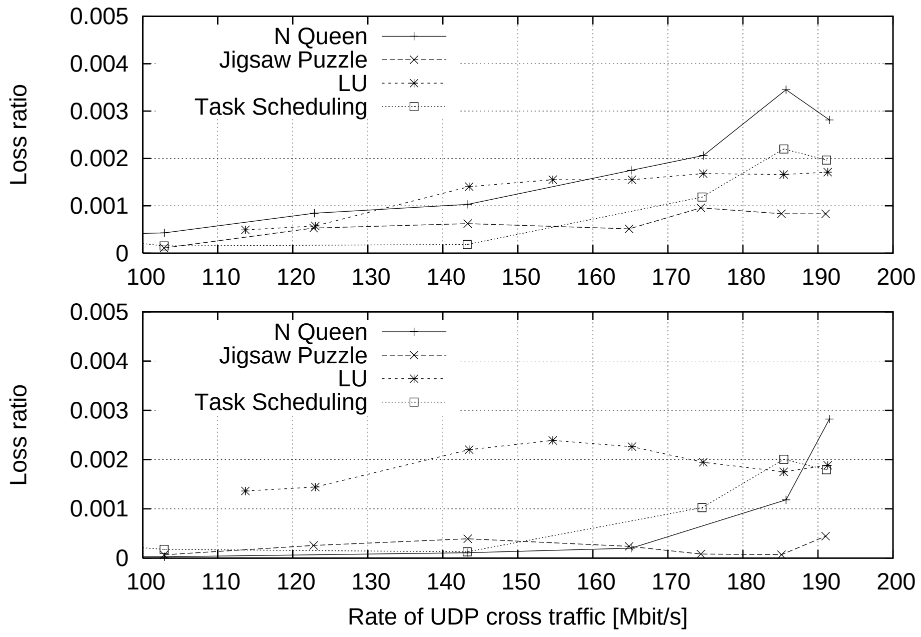


Figure 4.16: Packet loss ratio of UDP cross traffic sharing bottleneck link with application traffic

Figure 4.17 (4.18) shows the completion time for N Queen (LU Decomposition) when it is run with N Queen, Jigsaw Puzzle, LU Decomposition or Task Scheduling. The completion time in Figure 4.17 (4.18) is normalized by one when a single N Queen (LU Decomposition) runs under a 1-Gbit/s bottleneck link. The label ’single’ in Figure 4.17 (4.18) represents the completion time when single N Queen (LU Decomposition) is run in the given a RTT and a bandwidth of the bottleneck link, and ’double’ represents when double N Queens (LU Decompositions) are run simultaneously.

The order of the impacts of traffic generated by coexisting applications was the same (i.e., N Queen > Task Scheduling > LU Decomposition > Jigsaw Puzzle) for both applications, as showed in Figures 4.17 and 4.18, while the degree of the impacts differed depending on the application. N Queen influences on the coexisting application performance stronger than

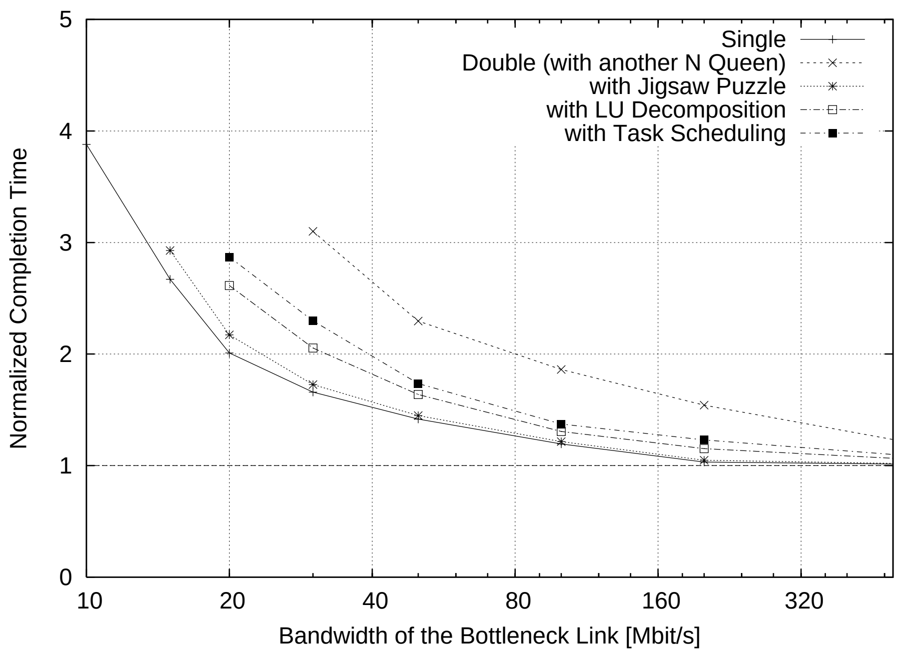


Figure 4.17: Completion time of N Queen influenced by other applications

other applications. In fact, N Queen requires more than double bandwidth with respect to the bottleneck link bandwidth to achieve the same level of performance as in the case of running a single application when two N Queens run simultaneously. This significant impact is probably because the throughput at the high rate generated from slaves to master influences on the coexisting application. There may be an opportunity that the traffic shaping[1]mitigates the effect of traffic of other applications on application-level performance.

The information on the impact of other applications on the application-level performance will be useful in determining which applications should be run simultaneously and which end-to-end path (of various alternatives) should be allocated to an individual application.

[1]Traffic shaping provides a mechanism to control the amount and volume of traffic being sent into to a link, and the rate at which the traffic is being sent.

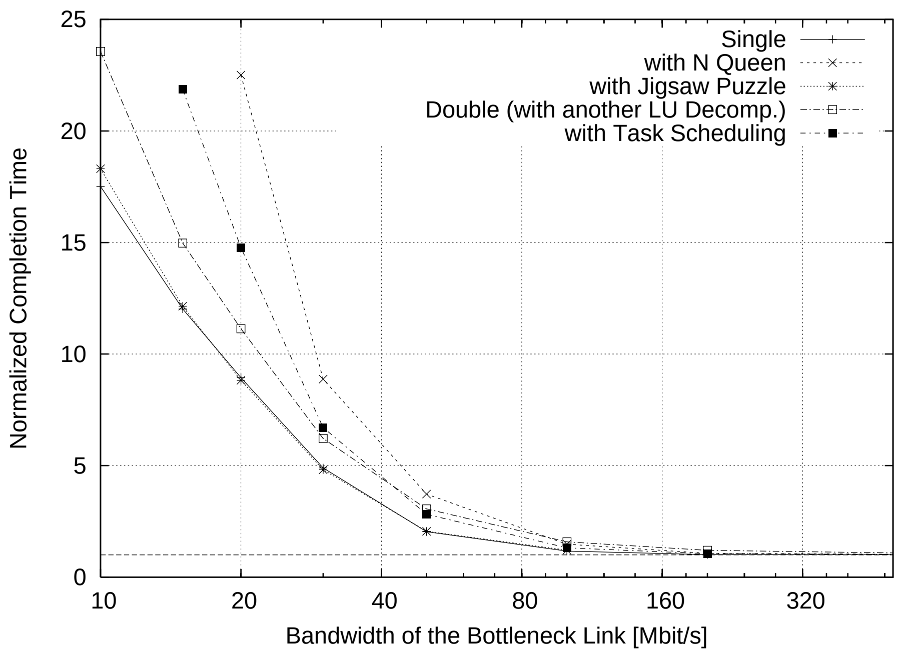


Figure 4.18: Completion time of LU Decomposition influenced by other applications

## 4.7 Example of Simple Traffic Engineering Based on the Application-Level Performance

It is showed that appropriate allocation of network resources (e.g., a bandwidth) contributes to application-level performance, and which thus indicates the effectiveness of using information on application-level performance as a basis for traffic engineering.

### 4.7.1 Example Scenario

A simple scenario of distributed computing over a dedicated network is considered (Figure 4.19). Every pair of PCs is connected to edge routers that are connected to aggregation routers that are also connected each other. Both routers are able to shape traffic. Two of the four applications illustrated in this chapter are simultaneously executed and each application

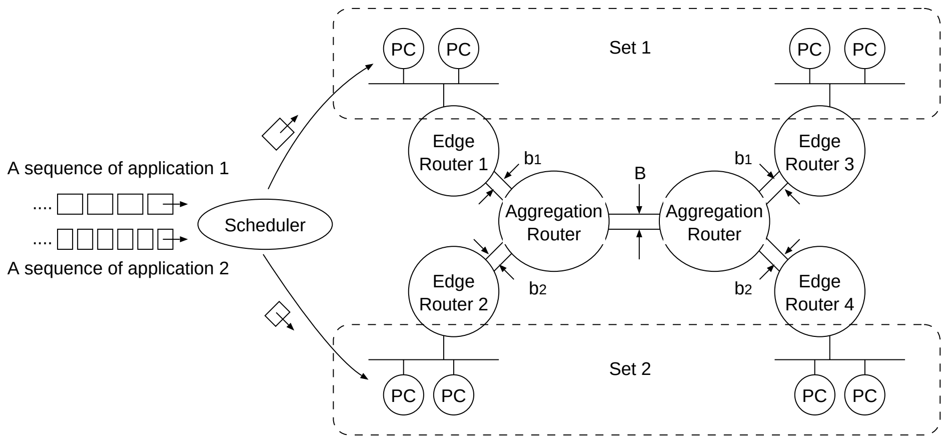


Figure 4.19: Network configuration for traffic engineering demonstration

is independently executed one after another infinitely. The application traffic therefore shares a bottleneck link between the aggregation routers continuously. Each application always runs on the same set of PCs (grouped by dotted lines in Figure 4.19).

Three types of scenario, as showed in Table 4.4, were conducted.

The link between the aggregation routers is assumed to be restricted up to bandwidth B. The bandwidth of local links connected to edge routers is also assumed to be greater than the bandwidth B of the bottleneck link. The edge routers are able to shape application traffic so that the total traffic for both applications does not exceed bandwidth B at the bottleneck if needed. The characteristics of the application-level performance, such as sensitivity of RTT, the threshold for limiting bandwidth, and the impact on other traffic were known in advance, and the CPU in the PCs could be fully utilized to process successive applications given these performance characteristics.

In this situation, intuitively, multiplexing the application traffic (not shaped at the edge routers; termed 'non-shaping case') is expected to produce higher application-level performance in total for two applications than shaping the application traffic at the edge routers (termed 'shaping case'). However, it is showed that the shaping case, so that total traffic does not exceed the bandwidth B, can achieve higher application-level performance using the performance characteristics.

### 4.7.2 Deriving Most Suitable Bandwidth

As the preparation, a performance function, which derives the completion time $p(x)$ $(> 0)$ for a given single application running under a condition of a bandwidth $x$ of the bottleneck link and a given RTT, is defined as follows:

$$p(x) = \frac{a}{(x^b - c)} + d, \quad \text{where} \quad x > c^{1/b}$$

where $a$, $b$, $c$, and $d$ are fitted using the least squares method so as to match the performance function with the target data. The parameters $a$, $b$, $c$, and $d$ differ depending on a pair of the target application and a RTT. E.g., the performance functions depending on the RTT in relation to the data for N Queen described in Figure 4.12 are showed in Figure 4.20.

In the demonstration, the total completion time, that is, a sum of the completion time for two applications running simultaneously, is considered. In the shaping case, where traffic from the edge router 1 (edge router 2) is shaped to $x_1$ ($x_2$), the total completion time $O(x_1, x_2)$ can be defined as follows:

$$O(x_1, x_2) = p_1(x_1) + p_2(x_2), \quad \text{where} \quad x_1 + x_2 \leq B \tag{4.1}$$

where $p_k(x_k)$ is the completion time of application $k$ $(k = 1, 2)$. $x_1$ and $x_2$ so as to minimize the total completion time, is chosen. These bandwidths is defined as the most suitable bandwidths.

### 4.7.3 Efficiency of Shaping Application Traffic Individually

The application performances obtained in between the non-shaping and shaping cases (Figure 4.19) under the conditions described in Section 4.7.1 are compared.

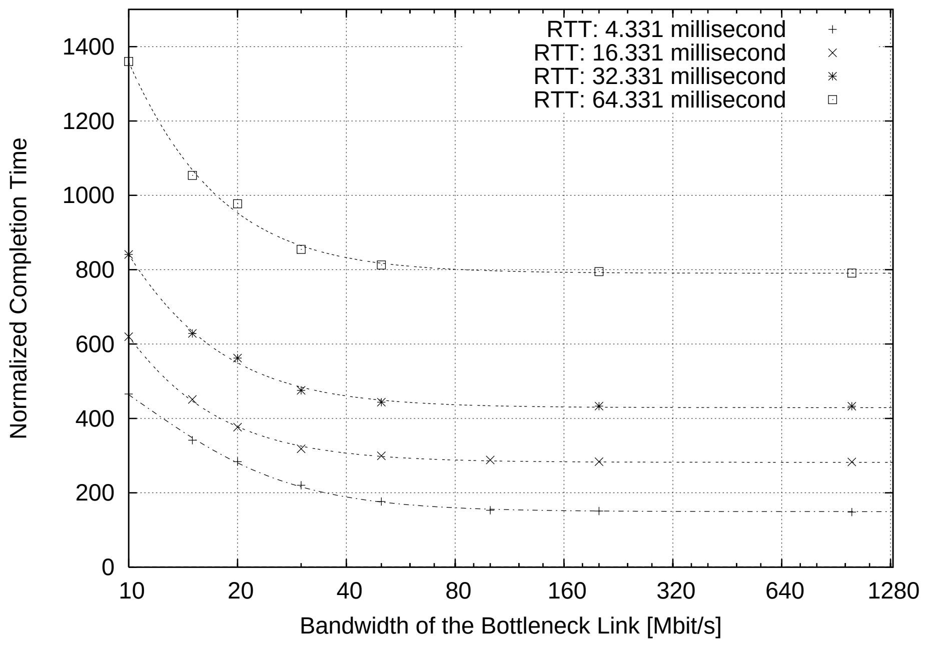


Figure 4.20: The completion time of N Queen and approximated performance functions

.

Table 4.4 shows the combination of applications, bottleneck bandwidth, and RTT of the path when the applications are run. Tables 4.5 and 4.6 show the each and total completion time of two application run in the non-shaping and shaping cases, respectively.

Comparisons of the performances in demonstration 1 shows exactly the same result as a comparison of the total completion time for a pair of a single-executed performance as in the shaping case and the completion time of a simultaneous performance as in the non-shaping case, as described in Figure 4.17. It is thus clear that in demonstration 1, better performances are produced in the shaping case rather than the non-shaping case.

With respect to demonstrations 2 and 3, to derive the most suitable bandwidth to allocate to each of the applications, the performance function is first fitted to the actual data which is, at this point, diverted from Section 4.5.3, as described in Section 4.7.2. For example,

Table 4.4: Demonstration scenario

| Demo. | Combination of applications | Bottleneck band width | RTT |
|---|---|---|---|
| 1 | N Queen & N Queen | 30 Mbit/s | 0.331 ms |
| 2 | N Queen & Jigsaw Puzzle | 20 Mbit/s | 4.3 ms |
| 3 | N Queen & Jigsaw Puzzle | 100 Mbit/s | 4.3 ms |

Table 4.5: Performance in non-shaping case

| Demo. | Application | Bottleneck bandwidth [Mbit/s] | Completion time [s] | Total completion time [s] |
|---|---|---|---|---|
| 1 | N Queen | 30.00 | 593.34 | 1186.67 ms |
| 2 | N Queen | 20.00 | 450.15 | 1010.15 ms |
|  | Jigsaw Puzzle |  | 560.00 |  |
| 3 | N Queen | 100.00 | 250.15 | 450.37 ms |
|  | Jigsaw Puzzle |  | 200.22 |  |

looking at demonstration 2, the performance functions for N Queen and Jigsaw Puzzle are respectively derived as follows[2]:

$$p_{(NQueen)}(x) = \frac{67543.039}{x^2 + 114.737} + 148.435$$

$$p_{(JigsawPuzzle)}(x) = \frac{72.7929}{x - 3.05654} + 62.0894$$

As a result of minimizing the total completion time described in Section 4.7.2, the most suitable bandwidth, as showed in Table 4.6, is theoretically derived.

Demonstrations 1 and 2 show that the total completion time in the shaping case is shorter than that of the non-shaping case, implying that the former method is more efficient than

[2]The unit of bandwidth is Mbit/s.

Table 4.6: Performance in shaping case

| Demo. | Application | Suitable bandwidth [Mbit/s] | Completion time [s] | Total completion time [s] |
|---|---|---|---|---|
| 1 | N Queen | 15.00 | 395.29 | 790.58 |
| 2 | N Queen | 14.38 | 592.16 | 906.61 |
| | Jigsaw Puzzle | 5.62 | 314.45 | |
| 3 | N Queen | 76.36 | 386.38 | 634.50 |
| | Jigsaw Puzzle | 23.64 | 248.12 | |

the latter. It was also observed that the performance was improved by shaping individual traffic under the condition that a short RTT and narrow bandwidth bottleneck link, and that non-shaping case was effective when there was a large RTT or enough bandwidth in the bottleneck.

Although the demonstrations described in this section are fairly simplified and the approximation of the performance function itself can be improved, essentially these demonstrations show that; (a) traffic engineering based on application-level performance is efficient, and (b) information on the impacts of network conditions on the application performance and that of application traffic on network conditions is vital for efficient traffic engineering under a range of network properties.

## 4.8 Conclusion

The network-related properties of some typical distributed applications, focusing on the influence of application traffic on the condition of network resources, and conversely, that of the condition of network resources on application-level performance were investigated.

First, the characteristics of traffic generated by applications classified as either task-framing or the work-flow types of distributed processing were analyzed. All the applications increased their completion time and the amounts of transmitted data as the scale of problems became larger while the relationship between the amount of transmitted data and the completion time depended heavily on the application.

The next analysis was how the performance of each application was affected by the condition of the network resources. It was found that the sensitivities of the completion time to the RTT and the bottleneck link bandwidth differed strongly depending on the application. Note that the availability of a large window in a TCP connection could mitigate the performance degradation caused by a long RTT in an application sending large amounts of data in succession. Furthermore, the influences of traffic generated by an application on other traffic sharing the bottleneck link and on the performance of other applications were analyzed. The results showed that the influences of traffic of an application on the loss ratio of UDP cross traffic and on the completion time of coexisting applications differed depending on the application. These differences should be considered in order to effectively allocate the network resources to applications simultaneously running on the network.

Finally, the examples that the information on the above mentioned network-related properties of individual application can be used to improve the total application-level performance, when two applications run through a shared link with a limited bandwidth, was showed. Thus, the followings were found: (a) the traffic engineering based on application-level performance was efficient, and (b) information on the impact of network conditions on application performance and the impact of application traffic on network conditions was vital for efficient traffic engineering with different network properties.

The goal of this study is to develop an application-aware method of allocating network resources using the information on the network-related properties of different applications.

The experiments indicate the feasibility of using information on application-level performance to allocate network resource efficiently.

# Chapter 5

# Performance Monitoring of VoIP Flows for Large Scale Network

## 5.1 Introduction

Due to advances in broadband Internet access, Internet service providers (ISPs) are now facing the challenge of providing reliable transport of real-time traffic, e.g., voice over IP (VoIP) and video conference. In Japan, 14% of home customers signing contracts with ISPs check if VoIP services are included and 43% of home customers use VoIP services [whi06]. So many ISPs now provide Internet accessibility for VoIP service, which includes interconnection with public switch telephone networks (PSTNs). ISPs must now seriously consider VoIP traffic performance when designing and operating their networks, since users generally expect VoIP service quality as high as that of PSTN service.

While the Internet has long carried various types of traffic (e-mail, Web, peer-to-peer [P2P]), VoIP traffic is quite different from previous traffic. However, it is difficult to identify degradation in VoIP performance from utility information of a link since there are different types of traffic on the link. For example, VoIP performance may be degraded due to burstiness of other traffic even if link utility is low. Therefore, the ability to monitor VoIP traffic exclusively is of significant importance.

Generally, first alert tools are used for ordinal network monitoring. They passively monitor traffic to detect signs of trouble, i.e., congestion, network attacks, or intrusions and alert network operators. Once an unusual situation isalerted to the network operators a variety of actions are taken with exclusive tools. Although the ISPs have so far been relying on link utility information, e.g., traffic rate variation monitored by MRTG [mrt], an exclusive first alert tool will now be required to monitor performance of VoIP traffic.

Estimating VoIP performance on boarder links connected to other ISPs is also important when VoIP service is provided through the interconnections. An ISP should know whether VoIP performance is degraded within its network or is already degraded when packets income from another ISP. They must monitor performance of VoIP traffic at ingress and egress to know whether VoIP performance is adequate within its network as well as its ingress connected to customers.

The key metrics of VoIP performance are delay variation and packet loss. The inter-packet gap (IPG) observed in individual VoIP flows are used to estimate delay variation. In this chapter, a flow is defined as as a sequence of packets distinguished by five tuples, that is, IP addresses, protocols and port numbers. For packet loss, detecting the missing packet, that is, counting the lack of numbered packets in individual flows by tracking application data in a packet, complicates packet processing. Because of this, VoIP flow performance monitoring systems often cannot deal with large amounts of VoIP flows. To solve this problem, the proposed method estimats delay variation and packet loss of VoIP flows by using the features of fixed packet length and almost fixed IPG.

The remainder of chapter is organized as follows. In Section 5.2, this approach is distinguished from the other ones. In Section 5.3, VoIP flow monitoring problems are clarified in terms of scalability. In Section 5.4, traffic traces obtained from Abilene [abi] and a certain ISP in Japan are analyzed the features of VoIP flows are identified. In Section 5.5, the proposed method is described how it extracts VoIP flows and estimates delay variation and packet loss on the basis of the features. In Section 6.5.4, the proposed method is evaluated by simulating packet arrival process with traffic traces. In Section 5.7, the proposed method is discussed on applying it to a real network monitoring operation. In Section 6.6, this chapter is summarized with concluding remarks.

## 5.2 Related Works

The conventional traffic monitoring is based on the utility information on the the network interfaces of routers or switches. The utility information basically shows the traffic rate averaged with a long period. The performance of VoIP flows may be wrongly understood when highly bursty flows are included in the same link as the VoIP flows and the average utility of the link is low [XG02] [TV00].

Netflow [net02] and sFlow [PPM01] are monitoring tools which produce high-level statistics on flows identified with five tuples. Since they are generally implemented in routers and switches, computing the statistics cannot obtain sufficient processing power in them. Therefore, monitoring deals with the limited number of flows or the severely inaccurate statistics are acquired due to packet sampling.

Calyptech produces an all-in-one system which non-intrusively monitors the quality of voice exchanged in various types of network, such as, VoIP, Wireless, PSTN, and enterprise networks [GT]. In the VoIP network case, their product, M10, collects VoIP packets with speed of up to 1 Gbit/s and analyze the mean opinion score using P.SEAM (ITU-T P.563 [p5604]) and E-Model (ITU-T G.107 [g1004]), echo return loss, periods of non-speech, noise, and so forth. Most metrics are for the voice quality degraded in terminal adapters or phones in a VoIP system and it doesn't directly monitor metrics representing the network properties affecting the VoIP flows. These metrics of significant importance when the problems are distinguished whether they come from network dynamic or devices used by customers. In addition, the complexity of analysis probably limits the number of flows that must be monitored in the backbone network.

Deri proposed the design and implementation of an open source tool for detecting and measuring VoIP traffic [Der06]. The tool collects NetFlow-like information on the VoIP flows with nProbe [Der03] and analyzes the VoIP metrics with ntop [DCS01] for understanding the network properties affecting the VoIP flows. The tool is targeted for monitoring various types of VoIP flows, such as, those with variable and fixed bit-rate and packet size. Such flexibility results in an obstacle to enabling the tool to deal with a large number of flows in a high-speed link, e.g. 10 Gbit/s.

This chapter emphasizes that in the ISP's network operation, the scalability for a large number of VoIP flows is more important than the flexibility to the various types of them. Therefore, a lightweight method for distinguishing VoIP packets for this scalability is proposed.

## 5.3 Performance Monitoring of VoIP Flows

### 5.3.1 Monitoring Locations in Backbone Network

Figure 5.1 shows the location of tools used to monitor VoIP flows in an ISP's backbone network. A monitor *A* is at the ingress link from customer edge router to monitor the degradation of VoIP flows in an area between the customers and the monitoring point. Other monitored areas are between *A* and *B*, *C*, or *D*. Degradation occurring in the backbone network is monitored in these areas.

Degradation in the core is more accurately inferred than in VoIP flow from customers because the characteristics of VoIP flows, such as, the IPG and the number of packets, monitored between two monitoring points can be compared and differences can be used to infer

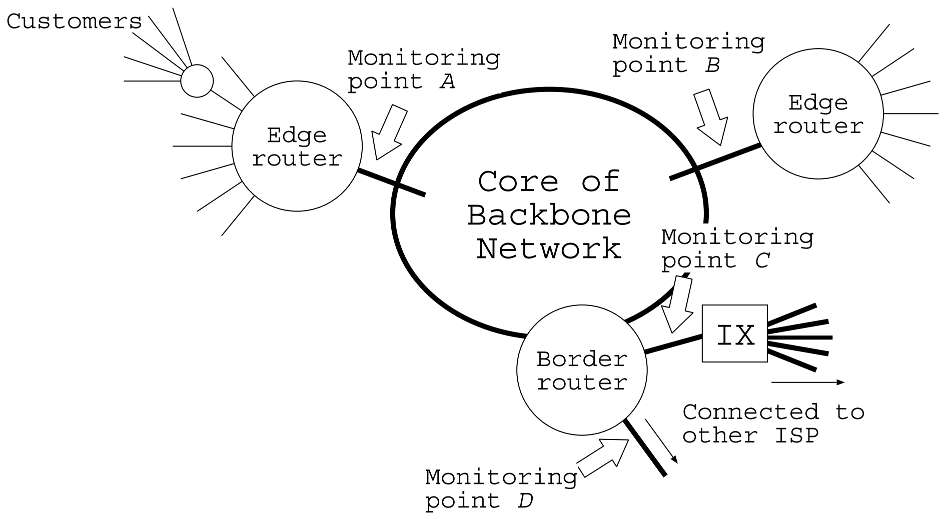


Figure 5.1: Location of monitoring tools in backbone network. An IX is an Internet exchange point.

degradation. In order to monitor VoIP flow from customers at a single point, the features of packet generation of a terminal adapter (described in Section 5.3.2), such as, the same inter-packet gap over packets in VoIP flows, should be used. Such use is the focus of this chapter. Section 5.4 demonstrates that the actual features of VoIP flows are usable.

### 5.3.2 Identification of VoIP Flows

A VoIP system is generally composed of an SIP server, VoIP terminal adapters (TAs), and phones. A call made over the Internet is set up in the following steps.

1. A caller TA requests a call setup on an SIP server with the information on the caller TA when a phone's receiver is taken off the hook and a number is dialed.
2. The SIP server announces the connection setup information on the caller TA, that is, an IP address and the dynamically assigned port number of

the caller TA to the callee TA, collects the information on the callee TA, and responds with the latter information to the caller TA.

3. The caller TA establishes a connection with the callee TA and a bidirectional VoIP flow begins between them.

As seen in this procedure, to monitor the VoIP flows, a monitoring tool needs to infer the port numbers of a VoIP flow, or to collect it from the VoIP entities. The SIP server and TAs can potentially notify the IP addresses and dynamically assigned port numbers for each of the VoIP flows. However, the SIP server or TAs and the monitoring tool do not share a common communication protocol. Therefore, it is assumed that without the information given by the VoIP entities, the monitor identifies VoIP flows.

There are four basic steps in monitoring the performance of VoIP flows:

1. capturing packets,
2. distinguishing VoIP packets from the observed ones,
3. identifying a flow from the VoIP packet using five tuples, and
4. estimating the metrics of the VoIP flow.

For Step 2), three types of method are generally used, a) port numbers in the UDP header, b) a flag marked by the TA, (i.e., a bit flag marked in an IP header), and c) attribute data used by the VoIP application. However, type a) cannot be used due to the above assumption. For type b), although the VoIP packets are distinguished without any errors, only few ISPs that can improve the TA they distribute to customers can use marking. Finally, identification using type c) is not always accurate because the attributes can be altered or concealed by VoIP applications [KSOT].

In this chapter, a method that distinguishes VoIP packets using the features of packets in VoIP flows is proposed. The proposed method can greatly reduce packet processing required to distinguishing VoIP packets. Therefore, it can be used for traffic level of 10-Gbit/s. Although the method can be applied only to limited VoIP flows from the VoIP application using specific codecs, It is showed how such a VoIP application can be easily adopted for VoIP services in ISPs in Section 5.4.

### 5.3.3 Metric Representing Degradation of VoIP Flows

The key metrics of VoIP performance are delay variation and rate of consecutive packet loss.

The monitor can exactly determine delay variations influencing the TA by emulating packet arrival process in a TA receiving a VoIP flow. However, such emulation is inadequate for monitoring performance in ISPs dealing with large amounts of VoIP flows due to scalability problem. Although the performance information conveyed by the extended report of the RTP header [SCFJ96] in a VoIP packet may be used, Section 5.4 shows that most VoIP packets do not carry the performance report (seen from the packet size). For these reasons, the variance of IPG showed by each VoIP flow is used as the delay variation. In addition, since IPG increases as the number of consecutive lost packets increases, a metric representing a difference of observd IPG from an ideal one is defined as the delay variation. For this, the ideal fixed IPG is used as well as observed IPGs. This also limits flow extraction to be used for the VoIP flows from the VoIP applications using specific codec. However, such a VoIP application is easily used in genaral is indicated in Section 5.4.

As a naive solution, tracking the packet sequence in a VoIP flow can determine the rate of consecutive packet loss, in the case of single point monitoring. Although the IP header has a sequence number, the packet number cannot be sequentially assigned to each packet of a VoIP flow when other flows occur simultaneously and use the same IP address as the VoIP flow. In addition, although the RTP header can have a sequence number, it is also conveyed in the option header and cannot be seen in most VoIP packets discussed in the VoIP flow analysis in Section 5.4. Therefore, a metric representing the rate of consecutive packet loss with the features of an almost fixed IPG, is also defined. This enables single point monitoring to determine the packet loss rate without tracking the sequence number in the packet headers. The metrics, the distance of IPG and rate of consecutive packet loss, are described in Section 5.5.1.

## 5.4 Characteristics of VoIP Flows

In this section, the intuitive characteristics of VoIP flows are described and then verified with traffic traces. Then, the best parameters for extracting VoIP flows is identified.

Intuitively, VoIP flows last a long time and have fixed packet length and fixed IPG due to the following reasons.

- A white paper [whi06] reported that average business and home calls last 39 and 92 seconds, respectively. Although the white the paper gives no statistics for VoIP flows, this result makes us to imagine that the duration of a VoIP flow should be on the order of tens of seconds.
- There should be a large number of VoIP flows with a fixed packet length, because it is likely that the G.711 (64 Kbit/s) codec [g7198] is adopted for most VoIP applications to enable a TA to interconnect to PSTNs, and because the increased availability of broadband Internet access has eliminated concerns about bandwidth use.
- To simplify the time-synchronization mechanism used to decode the digital signal from received packets, most TAs are designed to periodically generate packets, even when there is a silence for a while in a call.

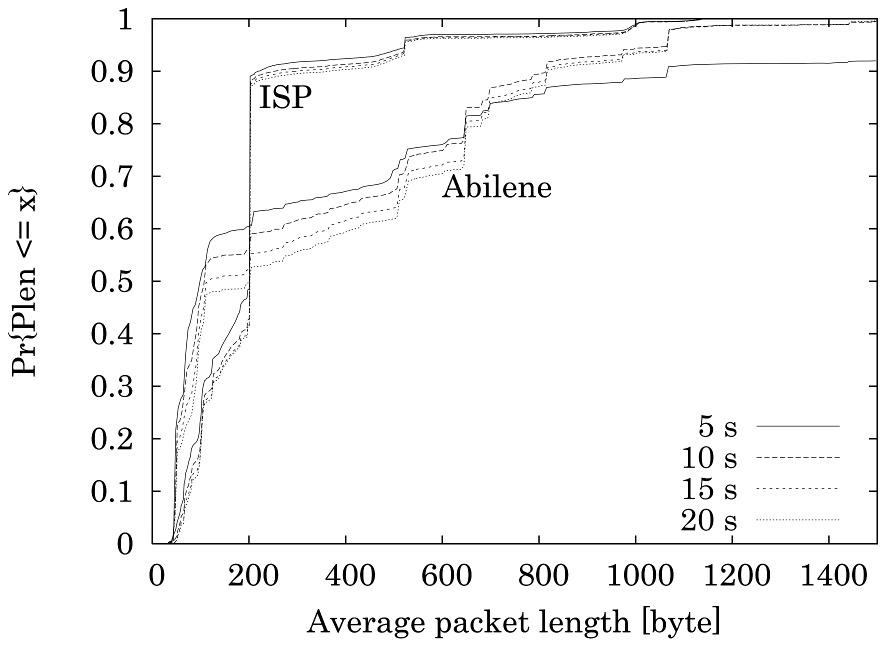


Figure 5.2: Cumulative distribution of average packet length for UDP flows of more than a certain duration.

To verify these suppositions actual traffic is analyzed. Figure 5.2 shows cumulative distributions of average packet length of individual UDP flows for the traffic traces collected from a 2.4-Gbit/s link in the Chicago PoP of Abilene in August 2002 and from a 1-Gbit/s link of an access point to the backbone of an ISP in Japan in July 2003. The traffic was fully captured for 100 seconds in the Abilene case, and 60 – 70 seconds in the ISP case (Table 5.1).

The distributions in the figure are for flows lasting more than certain durations (5, 10, 15, and 20 seconds). In the ISP case, the distributions for flows that last more than 10 – 20 seconds are similar. More than 45 % of the flows have an average packet length of 200 bytes. A packet encoded with the G.711 (64-Kbit/s) codec carries 160 bytes of voice data ($64000 \times 0.02/8$) assuming it is packetized every 0.02 seconds. Hence, with a 20-byte IP

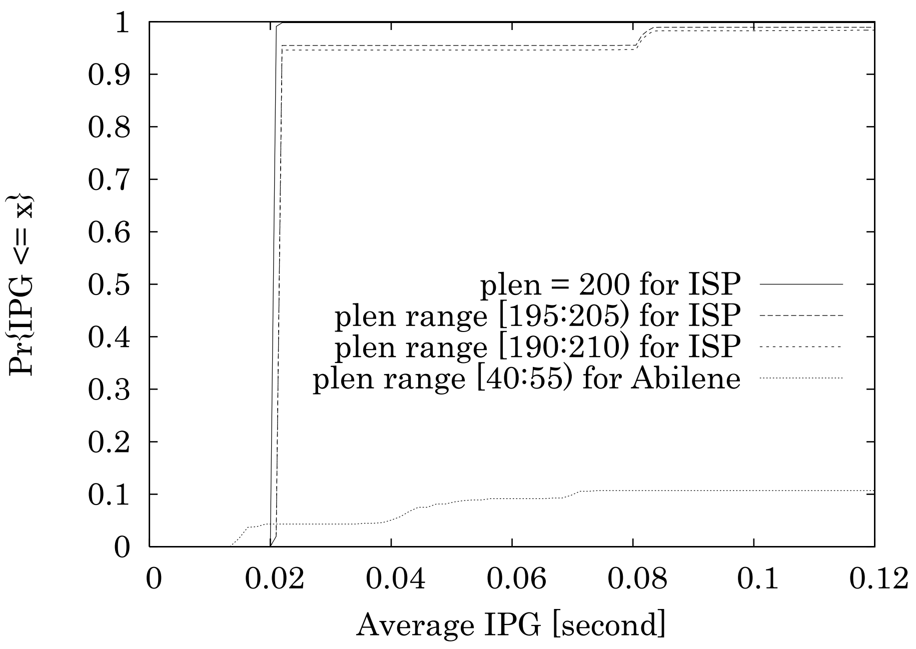


Figure 5.3: Cumulative distribution of average IPG for UDP flows lasting more than 10 seconds in a range of packet lengths.

header, an 8-byte UDP header, and a 12-byte RTP header, an IP packet is 200 bytes long. Therefore, flows with 200-byte packets are expected from TAs with the G.711 codec. For Abilene, the distributions are not similar to the ISP's. More than 20 % of the flows had an average packet length of 50 bytes. The IP packets may carry 10-byte voice data which comes from the G. 729 (8 Kbit/s) codec [g7296], if the voice data is packetized every 0.01 seconds.

To determine the relationship between codec and IPG (a packetizing interval), cumulative distributions of the average IPG for UDP flows of packets that last more than 10 seconds were analyzed. And it is found that they were about 200 or 50 bytes long for ISP and Abilene, respectively. Figure 5.3 shows the results of this analysis. The plots show that the 50-byte packets do not come from the G.729 codec because average IPGs have mostly unexpected values (more than half of the IPGs were about one second). So, the ISP's traffic traces is used

Table 5.1: Characteristics of traffic traces.

| Source | Length of data set (s) | Link bandwidth | No. of data sets | Total no. of packets |
|---|---|---|---|---|
| Abilene | 100 | 2.4 Gbit/s | 12 | 1.26 bill. |
| ISP | 60 – 70 | 1 Gbit/s | 55 | 175 mill. |

hereafter. It shows that more than 90 % of flows have an average IPG of about 0.02 seconds. Slight variations from 0.02 seconds are probably caused by the flows being attracted by delay variation.

Lastly, the duration of flows longer than 10 seconds with 200-byte-long packets and an IPG ranging from 0.01 to 0.03 seconds were analyzed (its figure is not presented). The average duration was 47 seconds and more than 75 % and 35 % of the flows lasted longer than 30 and 60 seconds, respectively. The feature of the long duration is expected to reduce frequency at which avtive VoIP flows is registered in the monitoring tool (see a pseudo code in Section 6.5),n long flows are easier for the proposed method to handle.

From these results, flows with 200-byte-long packets and an average IPG of around 0.02 seconds in are used in the proposed method. Many VoIP applications currently generate packets of different lengths and IPGs. However, it is likely that an VoIP application provided by not only this ISP from which the traffic trace is collected but also each of the other ISPs will have the similar features in their VoIP flows, that is, the fixed-length packets and a fixed rate packet generation for the reasons described earlier in this section. Therefore, the proposed method will be easily applied merely by adjusting conditions of the packet length and IPG.

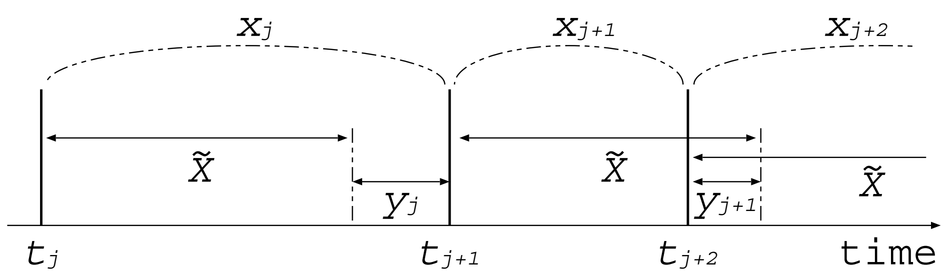


Figure 5.4: Packet arrivial process for flow *i*.

## 5.5 Proposed Method

### 5.5.1 Metric Definition for Delay Variation and Rate of Consecutive Packet Loss

As described in Section 6.3.2, two metrics are defined for the rate of consecutive packet loss and variance of IPG representing performance of VoIP flows. The IPGs are collected from all of the packets of $N$ sampled flows observed during a monitoring period (1 through 10 senconds) at a monitoring point. When an IPG sufficiently longer than an ideal IPG (based on the features of VoIP flows described in Section 5.4), is found, it is imagined that there must be a single or consecutive multiple packet loss. a difference from the ideal IPG which rises when there are lost packets is corrected as if there are an adequate number of packets in the long IPG.

Let $t^i_j$ denote time when packet $j$ in flow $i$ passes the monitoring point, and $x^i_j$ denote the IPG between the $j$-th and $(j+1)$-th packets (Figure 5.4). A *difference from ideal IPG* $\tilde{X}$, denoted by $y^i_j$ $(> 0)$, for an IPG $x^i_j$ is defined as a metric representing the delay variation. To define $y^i_j$, a factor of the ideal IPG, denoted by $k^i_j (= \max(1, \lfloor x^i_j/\tilde{X} + 1/2 \rfloor))$ are used. It satisfies

$$k_j^i = \begin{cases} 1 & \text{if} \quad 0 \leq x_j^i < 3\tilde{X}/2 \\ m & (m = 2, 3, \cdots) \\ & \text{if} \quad (m - 1/2)\tilde{X} \leq x_j^i < (m + 1/2)\tilde{X}, \end{cases} \tag{5.1}$$

resulting in

$$y_j^i = |x_j^i - k_j^i \tilde{X}| \, . \tag{5.2}$$

With an indicator function

$$\chi_{k_j^i \geq 2} = \begin{cases} 0 & \text{if} \quad 0 \leq k_j^i \leq 1 \\ 1 & \text{if} \quad k_j^i \geq 2 \, , \end{cases} \tag{5.3}$$

the rate of large IPGs treated as packet loss ($x_j^i \geq 3\tilde{X}/2$) is represented as

$$r = \frac{\sum_{i,j} \chi_{k_j^i \geq 2}}{N \sum_i n_i} \, , \tag{5.4}$$

where $N$ and $n_i$ are the number of sampled flows and the number of packets in flow $i$, respectively. So, $k_j^i$ is here called an inferred loss.

$r$ presents the rate of actual consecutive packet loss when the buffer size of all links through which the monitored packets passed is $\tilde{X}/2$ in time, or the rate of a combination of packet loss and IPGs exceeding $3\tilde{X}/2$ when the buffer size is larger than $\tilde{X}/2$ for the links. In the latter case, the monitor counts packet loss despite that there is actually no packet loss when the IPG increases significantly. However, when the IPG exceeds $3\tilde{X}/2$, the receiving TA probably must experience quality degradation as if the packet loss occurs for following two reasons. A large IPG is counted as a lost packet by the receiving TA if it has a small buffer size. And, a small buffer size in TA is recommended [ets02]. Therefore, $r$ is defined as the estimation of the rate of consecutive packet loss in terms of VoIP flow quality. Hereafter, *degraded packet rate* $r$ is called a rate of consecutive packet loss. With this estimation, the packet loss rate can be estimated even with single point monitoring.

Variance of the pseudo IPGs, denoted by $S^2$, is defined as delay variation.

$$S^2 = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{n_i}\sum_{j=1}^{n} {y_j^i}^2 . \tag{5.5}$$

This estimates the delay variation from which the impact of packet loss is removed when there is a packet loss. Hereafter, $S$ is used to term *average difference* from the ideal IPGs for the standard deviation of actual IPGs.

### 5.5.2 Algorithm to Register Monitored VoIP Flows

From the observation described in Section 5.4, it is found that there were a sufficiently large number of flows with packets of 200 bytes and an average IPG of around 0.02 seconds. Therefore, the method extracts flows having fixed packet length, almost fixed IPG, and a certain flow duration for performance monitoring. In the ISP case described in Section 5.4, the fixed packet length is 200 bytes, the fixed IPG is 0.02 s ( $= \tilde{X}$), and the minimum flow duration is 10 seconds.

Figure 5.9 shows a pseudo code used to identify monitored VoIP flows. The algorithm greedily registers VoIP flows in set $\mathcal{M}$, until the number of registered flows reaches a given number $N$. The identified flows lasting less than 10 seconds are registered in set $\mathcal{P}$. When a packet arrives, the algorithm first check if the packet is UDP and packet length, and identifies a flow of packets satisfying these conditions. The flow identification is inevitable due to obtaining the IPG based on a flow. When it is UDP and required packet length, the IPG variance is computed as described in Section 5.5.1 if the flow is already registered in set $\mathcal{M}$. A flow is moved to set $\mathcal{M}$ from $\mathcal{P}$, if the flow is found in $\mathcal{P}$ and lasts more than the required minimum flow duration. The algorithm also periodically monitors the termination of flows registered in sets $\mathcal{M}$ and $\mathcal{P}$, and unregister them if they terminate.

## 5.6 Performance Evaluation

### 5.6.1 Accuracy

The proposed method is evaluated, first, how well the degraded packet rate and the average difference of IPGs agreed with the actual the packet loss rate and the actual IPG, respectively, and, second, how the metrics obtained from sampled flows accurately present that from all flows. This is conducted by simulating a virtual link through which ISP's traffic traces used in Section 5.4 and virtual VoIP flows with 200-byte packet length and an ideal IPG of 0.02-second (termed *ideal flow*) are forwarded. The virtual link has a particular buffer size and a particular bandwidth in order to make the queuing delay of packets happen.

First, the accuracy of the degraded packet rate and the average difference of IPGs are forcused on. The evaluation shows the differences of them from the actual consecutive packet loss rate and the standard deviation of actual IPG of them, respectively. Figure 5.5 shows a Q–Q plot of the actual packet loss rate for all inserted flows and the degraded packet rate of sampled flows for seven sets of traffic traces when the buffer size was limited to that at which maximum queuing delay was 10 ms. In this case, 20 – 500 flows were used in the monitoring, and a thousand flows (80 Mbit/s in total) were inserted. The degraded packet rate is considered to represent the actual packet loss for the monitored flows because the buffer size is congruent with one used for the inferred loss. The virtual link was shaped to 381 Mbit/s, resulting in a total utility of 95 % (the traffic traces averaged 282 Mbit/s, and the ideal flows amounted to 80 Mbit/s). The traffic traces used changed more rapidly than traffic in any other sets.

The degraded packet rate is not very accurate. This is because the proposed method cannot detect the loss in all of the inserted flows when a small number of flows are sampled. When the proposed method estimates packet loss rate (> 0.2 %) the results are likely to

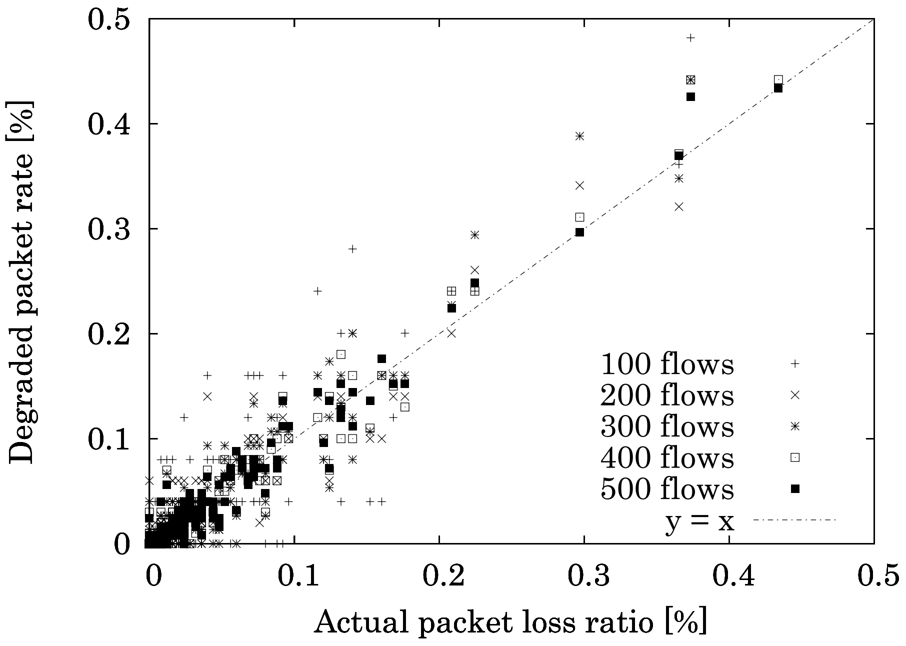


Figure 5.5: Q-Q plot of actual packet loss rate for all inserted flows (X axis) and degraded packet rate of sampled flows (Y axis) when buffer size was limited to that at which maximum queuing delay of 10 ms.

overestimate. Additional experiments with 500, 2000, and 5000 ideal flows (figures are not presented), showed that accurate packet loss estimation required the monitoring of at least 20 – 50 % of all ideal flows and the estimation included a 0.2 % error. This implies that in the case of the active measurement with probes emulating VoIP flows, huge amounts of probes are needed to be inserted into the monitored link to estimate fairly accurate rate of the consecutive packet loss.

Therefore, the passive measurement suits for monitoring performance of VoIP flows more than the active measurement.

Figure 5.6 shows Q – Q plots of the standard deviation of actual IPGs and the average difference of IPGs for all inserted flows. The conditions for selected traffic trace, the number

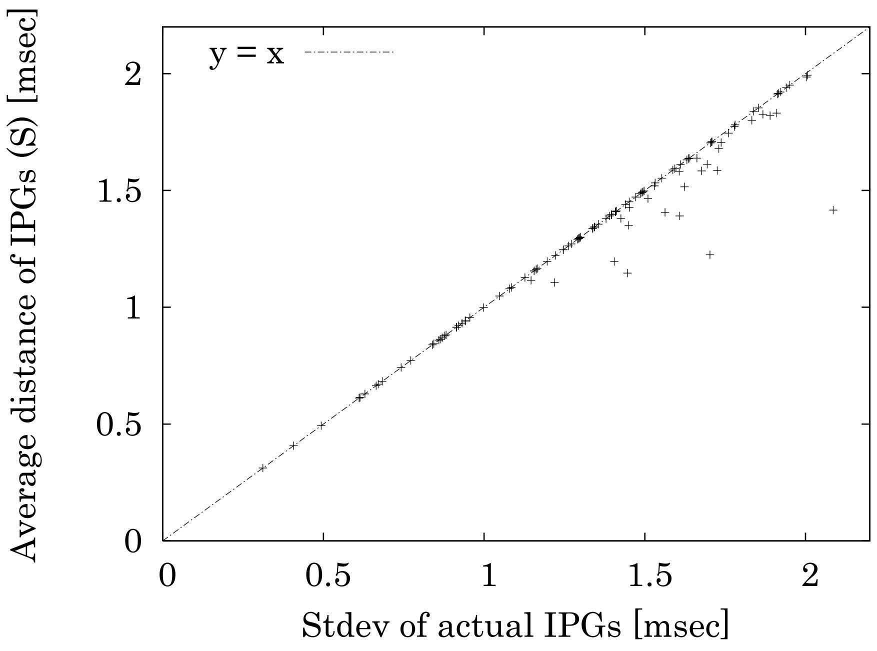


Figure 5.6: Q-Q plots of standard deviation of actual IPGs (X axis) and the average difference of IPGs (Y axis) for all inserted flows.

of the inserted ideal flows, buffer size, and bandwidth of the virtual link are the same as for the evaluation of the degraded packet rate. The figure shows that the average differences are lower than the standard deviation of the actual IPGs. This is because the difference from the ideal IPG is reduced when packet loss is detected.

Figure 5.7 shows Q – Q plots of the average difference of IPGs for all inserted flows and for sampled flows. The figure shows that the average difference of sampled flows is fairly congruent with that of all flows even when only 20 flows (2 % of the total) were sampled. When estimations were very inaccurate, it was usually due to rapid changes in traffic volume. In this case, the change should be seen in the degraded packet rate.

The results prove that the degraded packet rate requires large amounts of sampled flows to estimate relatively accurate consecutive packet loss and the average difference of IPGs for

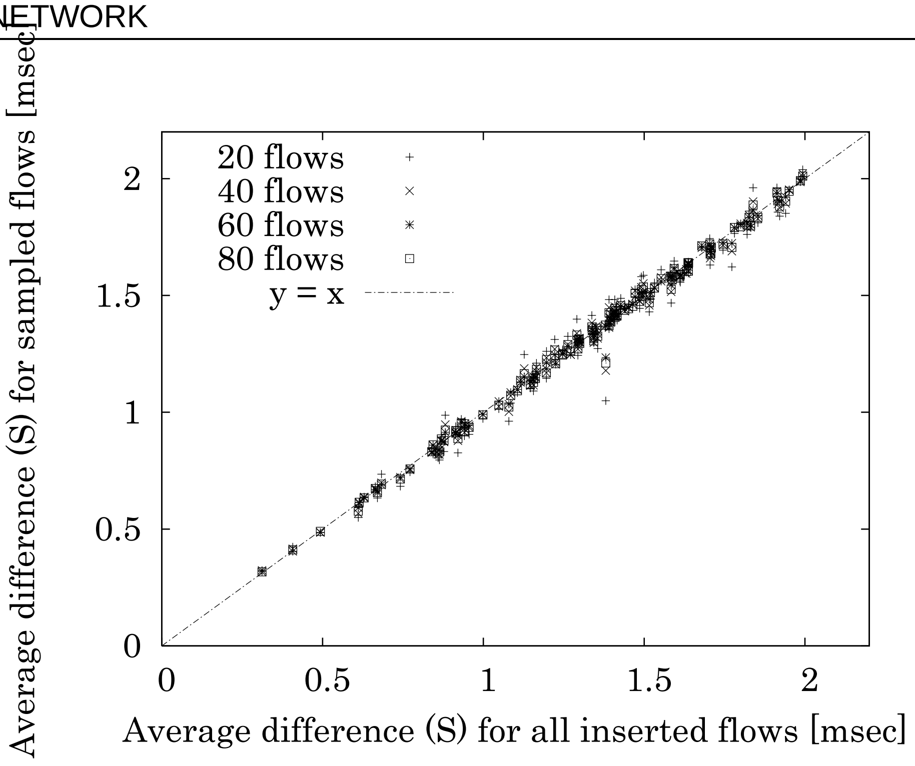


Figure 5.7: Q-Q plots of the average difference of IPGs for all inserted flows (X axis) and for sampled flows (Y axis).

the sampled flows is a sufficiently accurate reflection of delay variation in the VoIP flows and can be obtained by monitoring small amounts of sampled flows. Their reasons are discussed in the following section.

### 5.6.2 Number of Monitored Flows

For the degraded packet rate, The analysis showed that the number of sampled flows *N* should be sufficiently large. The reason can be seen from the consideration that the number of packet of sampled flows appeared in each period of ideal IPG. In every period, the number of degraded packet among the *N* flows follows the binomial distribution under the assumption that a single packet is showed from each flow in each period Due to a central limit theorem, this results in the larger number of monitored flows making the degraded packet

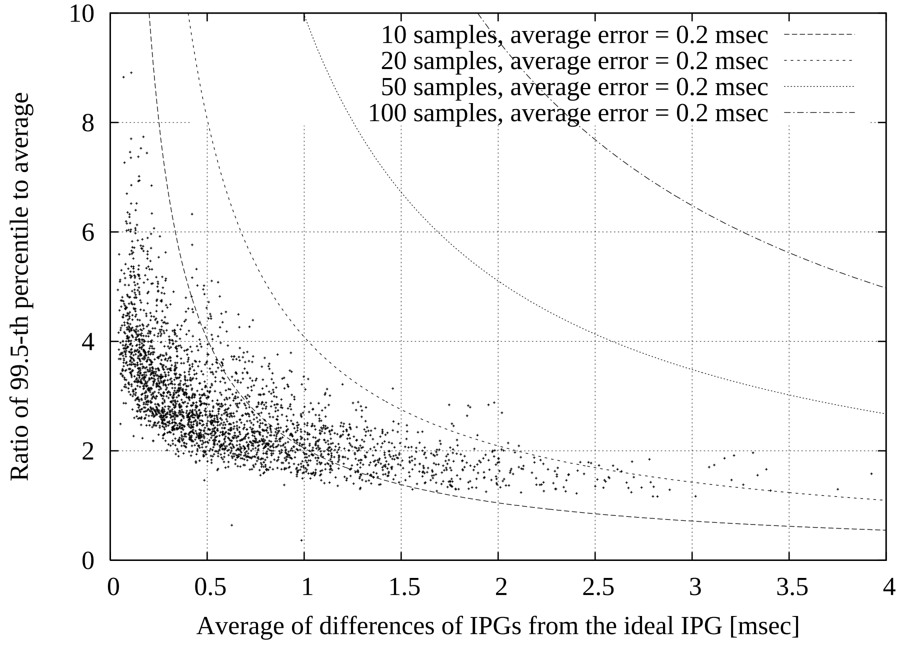


Figure 5.8: Q-Q plot on relationship between average and 99.5th percentile of differences from the ideal IPG in each period of ideal IPG (0.02 s).

rate converged the real rate.

Figure 5.8 illustrates that small amounts of sampled flows give the sufficient accuracy in computing the average difference of IPGs. The figure shows relationship (points) between the average and 99.5th percentile of the difference in each period (the ideal IPG). The Y axis presents the ratio of the 99.5th percentile to the average. This result used 0.01 s of the buffer size in time for the simulation and inserted VoIP flows by 10% of total traffic of trace. The points show that the results with the higher average is the smaller variance it has. The actual error is expected to be limited by a certain value due to the large average having small the 99.5th percentile.

The curves in the figure present this expectation. The curve is the limit of the 99.5th percentile, to let the average error of the average distance from a certain number of sampled flows to be 0.2 milli-seconds. These curves are computed under the assumption that each distance from the ideal IPG of each of 20 – 100 sampled flows gives the actual average distance and 99.5$^{\text{th}}$ percentile distance with probability of 0.995 and 0.005, respectively. The points with smaller 99.5$^{\text{th}}$ percentile than the N-sampled-flow curve mean that average error of the average distance introduced by sampling N flows compared with all the flows is expected less than 0.2 ms. With comparison the points and the curves, the figure shows that the errors in computing the average distance of IPGs with 20-sampled flows are almost less than 0.2 ms and exceeds it in a few cases. This indicates that even the small amount of sampled flows gives the sufficient accuracy as indicated in Section 5.6.1.

## 5.7 Applying Proposed Method to Real Monitoring Operation

For the proposed method, the fixed packet length and the ideal IPG are needed to distinguish VoIP packets and to compute the metrics of VoIP performance degradation a priori. These should be analyzed as showed in Section 5.4 before operating the monitoring. If the ISP distributes TA to customers, the packetizing rate and codec with which the packet length and the IPG are derived can be known a priori.

To calculate average difference of IPGs and degraded packet rate, $N$ flows are sampled from all the VoIP flows. Fewer flows ease computation of the metrics in the case of huge amounts of VoIP flows are active in a monitored link.

The proposed flow registration algorithm (described in Section 6.5) can be applied to a high-speed link, e.g., 10 Gbit/s, for the following reasons. Extracting 200-byte packet can reduce the number of processed packets. From the another analysis, it is found that UDP packets accounted for only 5.3 and 5.2 % of packets in the Abilene and ISP traffic traces, respectively. 200-byte packets represented 7.9 and 60.9 % of the total number of UDP packets for Abilene and ISP, respectively. Hence, 200-byte UDP packets accounted for less than 4 % of the total even in the ISP's traffic traces. This ratio is sufficiently low for the algorithm to monitor VoIP flows even in a 10-Gbit/s link. Even if VoIP flows become more prevalent, e.g., half of all the flows, because of the proposed algorithm's simplicity, it should be easy to implement in hardware to scale for larger numbers of VoIP flows. For estimation of the degraded packet rate to be sufficiently accurate with the proposed method, a maximum of 100 % of the VoIP flows may need to be monitored. Even in this unlikely situation, that is, if a 10-Gbit/s link were filled only by VoIP flows (156K flows), the recent LSI technology would enable monitoring of the flow even with such large amounts of flows.

Even if packet sampling is introduced before VoIP packets are distinguished to reduce the number of packets that must be treated, the method can still estimate delay variation. In this case, un-sampled packets are treated as consecutive packet loss and $k_j^i$ is frequently larger than one. Therefore, the degraded packet rate becomes heavily higher than the actual packet loss rate. However, delay variation can still be estimated. Note that identifying flows is inevitable in the proposed method, the packet sampling rate should be some amount so that the flows in set $\mathcal{M}$ and $\mathcal{P}$ are kept.

The comparison of metrics computed by two monitoring tools estimates the degradation level between two points. The comparison of the rates of the consecutive packet loss acquired from the different flows must largely differ from that from the same flow between two points. Therefore, the flows monitored at two points should be synchronized to make estimation of degraded packet rate more accurate. Developing an actual method to synchronize the monitored VoIP flows remains as future work.

## 5.8 Conclusions

A method for extracting VoIP flows from Internet traffic in a lightweight manner by using knowledge of the specific features of VoIP flows, e.g., the fixed packet size and long duration, are proposed. The proposed method can distinguish VoIP flows in a high-speed link, e.g., 10 Gbit/s. And a method of estimating the delay variation and packet loss ratio using characteristics of VoIP flows, i.e., inter-packet gap (IPG) is constant for VoIP flows, is also proposed simultaneously. Evaluation using simulation with actual traffic has showed that delay variation (IPG variance) can be accurately estimated by monitoring only few percentage of all flows. In contrast, estimation of the packet loss rate with an error of around 0.2% requires that large amounts (hundreds) of flows be monitored. Additionally, with the proposed method, packet loss rate can be estimated using single-point monitoring. It is, therefore, feasible to estimate VoIP performance by monitoring IPGs of a few flows, whereby ISPs will be able to assess whether or not VoIP service quality is being seriously degraded. It is for further study on the details of how to implement the proposed method, e.g., harmonization of the obtained data between ingress and egress points.

```
Function MonitoringFlow;
var $N$: Number of flows to be monitored;
var $\tilde{X}$: Expected average IPG;
var $L_{min}$: Minimum packet length;
var $L_{max}$: Maximum packet length;
var $T$: Threshold to register flows;

begin
  var $\mathcal{M} := \emptyset$: set of monitored flow;
  var $\mathcal{P} := \emptyset$: set of active flow w/t short duration;
  var $\mathcal{F} := \emptyset$: set of flow information base (FIB);
  repeat
    var $p := ArrivingPacket(L_{min}, L_{max})$;
    var $f := FlowID(p)$;
    $UpdateFIB(p, f, \mathcal{F})$;
    if $f \in \mathcal{M}$ then
      $UpdateStatistics(f, \mathcal{M}, \mathcal{F})$;
    else
      if $f \in \mathcal{P}$ then
        if $FlowDuration(f, \mathcal{F}) \geq T$ then
          $RemoveFlow(f, \mathcal{P})$;
          if $\tilde{X}/2 \leq IPG(f) < 3\tilde{X}/2$ then
            $InsertFlow(f, \mathcal{M})$;
            $UpdateStatistics(f, \mathcal{M}, \mathcal{F})$;
          endif;
        else;
          $AccountPacketAndByte(p, f, \mathcal{F})$;
        endif;
      else
        if $|\mathcal{M}| + |\mathcal{P}| < N$ then
          $InsertFlow(f, \mathcal{P})$;
        endif;
      endif;
    endif;
    $RemoveInactiveFlow(\mathcal{M}, \mathcal{P}, \mathcal{F})$;
  until receiving a signal to terminate;
end;
```

Figure 5.9: Flow monitoring algorithm. Parameters, $N$, $\tilde{X}$, $[L_{min}, L_{max})$, and $T$, are set to, for example, 100, 0.02 seconds, 200 or 201 bytes, and 10 seconds.

# Chapter 6

# Traffic Characteristics-aware Flow Assignment Method for Reducing Queuing Delay

## 6.1 Introduction

Internet service providers are facing the challenge of dynamically and adaptively designing their networks to satisfy customer demands for fast, reliable, and differentiated services with the minimal cost. Internet traffic engineering (TE) [ACE+02] is a key tool for meeting this challenge because it can effectively map traffic demands onto the network topology and adaptively reconfigure the mapping as network conditions change. More specifically, TE can adaptively distribute traffic flows over networks so that network resources are efficiently allocated.

The fundamental philosophy for making the Internet scalable is that an algorithmically complex processing should be pushed to the edge of the network whenever possible. When designing a backbone network that supports TE the backbone network having two levels of components – a high-speed core and edge routers surrounding the core – reflects this philosophy. Consider the case in which individual pairs of edge routers establish multiple paths to convey traffic from one edge (ingress) router to another edge (egress) router. Effective traffic distribution can be achieved by balancing traffic flows among these paths [EJLW01] [GKL+04].

In distributing the traffic flows, however, inadequate traffic balancing into multiple paths significantly degrades queuing delay performance, especially when the path is highly utilized [TV00] [XG02]. When different (bursty and non-bursty) types of flows are given to be balanced over multiple paths, the non-bursty flows assigned to the same path as the bursty flows encounter the same queuing delay performance as the bursty ones. In addition, when the path bandwidths are different, conventional path utilization based flow assignment, that is, assigning flows on the basis of the ratio of path bandwidths for each flow type, results in dropping the queuing delay performance of the narrower path more severely. Therefore, given a set of flows with distinct traffic characteristics and different path bandwidths, it is not trivial to determine how to distribute them so as to minimize overall queuing delay, that is, to achieve a good balance of the flows in terms of their queuing delays.

In this chapter, a simple case is considered. That is a number of flows having one of two distinct traffic characteristics, are assigned to one of two paths with different bandwidths (Figure 6.1). A queuing delay statistic, such as, the mean queuing delay, the standard deviation of the queuing delay, or the 99.5th percentile queuing delay for the worse path, is used as a metric representing the overall delay performance. The flows are assigned using the min-max fairness optimization of the queuing delay. It is showed that a flow assignment method improves this delay metric, and the queuing delay obtained from an flow assignment estimated by the proposed method is fairly lower than that from the conventional path utilization based flow assignment. The flow are defined as a sequence of packets distinguished by five tuples (a pair of IP addresses, a protocol number, and port numbers), however the propose method limits flows to only this definition. The method is applicable to *flows* in which

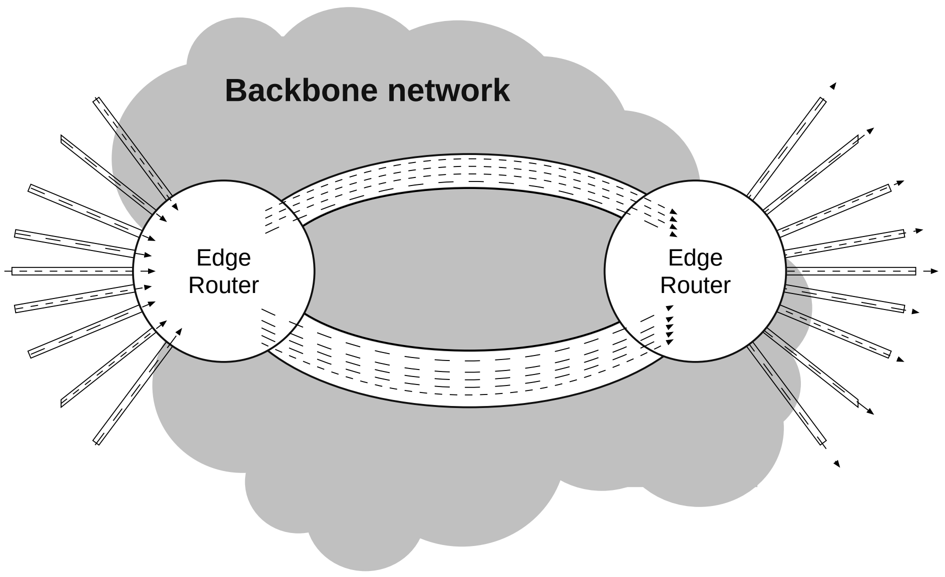


Figure 6.1: Simple case considered in this chapter: a number of flows, each having one of two distinct sets of traffic characteristics, are assigned to one of two paths with different bandwidths.

1. each packet can be promptly categorized on the basis of information it contains, and
2. each flow can be categorized by traffic types on the basis of its measured or predefined traffic characteristics.

The method is showed to be practical for roughly estimating the queuing delay statistics for an arbitrary assignment of two given types of flows to two given paths where traffic characteristic of each flow type and bandwidth of each path (each output link) are known. It learns the relationship between the flow assignment and the queuing delay statistics from the queuing delay statistics for a small number of previous flow assignments. Using this relationship, it estimates the queuing delay statistics for a period short enough to be used in an online manner, which is not possible with either fluid-flow model-based analysis nor packet-based simulation. Using the estimated statistics, it searches, in an online manner, for the better flow distribution over the two paths in terms of overall queuing delay performance to determine which path should be assigned to the next incoming flow. The effectiveness of flow assignment on the basis of the traffic characteristics of each flow showed through preliminary experimental evaluation demonstrates the great potential of this method.

In this chapter, queuing delay statistics, such as the mean, standard deviation and $99.5^{\text{th}}$ percentile of the queuing delay, are used in defining the delay metric for overall delay performance in assigning some flows to multiple paths. First, the delay metrics representing the delay performance for two paths are defined in Section 6.3.2. And next, the mean, standard deviation, and $99.5^{\text{th}}$ percentile of queuing delay encountered by two distinct flows assigned to a single path from the fluid flow model are derived in Section 6.3.3.

The rest of this chapter is organized as follows. Section 6.2 distinguishes this study's approach from previous ones. In Section 6.3, the flow assignment and its objectives are described. In Section 6.4, analysis of assigning two distinct flow sets to two paths in terms of the queuing delay, shows how effective traffic characteristics-aware flow assignment is. In Section 6.5, the proposed method is described and evaluated. Finally, Section 6.6 concludes with a brief summary.

## 6.2 Related Work

The efficiency and reliability of network operations can be improved with dynamic reconfigurations of physical (optical transport) and logical networks, and a traffic assignment to logical network at every instance of traffic condition. The one of most efficiency by the physical and logical network reconfiguration is making the reachability reliable [PDC+06] [Son01]. The logical network allows edge routers to have multiple paths spatially routed over the physical network, which enables the network to be adequately utilized as a whole. The cooperation of the traffic assignment with the logical network reconfiguration delivers the opportunity of traffic treatment to prevent delay performance degradation besides to improve the network utilization. This study focuses on the flow assignment methods.

The previous work proposed admission control algorithms that treated how the tunnel paths were routed between two edge routers [KKL00] [GCL04a]. These algorithms search for a path that minimizes the failure of the forthcoming tunnel establishment. They are categorized into the reconfiguring the logical network and It is assumed with the algorithms that the bandwidth required for the tunnel aggregating the forthcoming arriving flows could be predicted in an a priori manner. However, such a prediction often leads to bandwidth over-allocation to prevent congestion. The approach can be applied to such tunnel links to meet the given delay requirements in advance.

Recently, an advanced admission control algorithm that addresses bandwidth allocation for a tunnel by introducing the probability of violating the delay tolerance with deterministic delay bounds was proposed [GCL04b]. This leads to reconfiguration of the logical networks. The algorithm improved the utilization of a base link admitting tunnels while satisfying the delay requirements as well as the average throughput. However, the algorithm had a scalability problem in admitting flows with various traffic characteristics because individual tunnels were for homogeneous traffic characteristics. In contrast, the principal challenge in this study is on heterogeneous flows admitted in a path.

In assigning traffic to multiple paths, traffic can be distributed by balancing traffic flows between a pair of edge routers among several paths [EJLW01] [GKL+04]. The basic concept here is that the queuing delay is lowered by reducing path utilization. However, the low queuing delay is retained even when the utilization becomes high by taking the traffic characteristics into account, as Section 6.4 presents. The proposed method improves the delay performance by introducing delay characteristics on the multiplexed flow into the flow assignment.

## 6.3 Flow Assignment Problem and Analytical Model

### 6.3.1 Approach

It is assumed that the core of a high-speed network accepts bandwidth-guaranteed paths established between pairs of edge routers. Each pair establishes two or more paths. The traffic is distributed in a flow-based manner instead of a packet-based one which degrades the performance of TCP data transmission and the stability of UDP streams due to packet reordering.

The procedure of balancing flows over multiple paths at the edge router is

1. first grouping on-going flows into multiple types based on their traffic characteristics (peak rate, average rate, traffic-varying interval [cycle of on-off-states]) in an online manner, and
2. secondly distributing the flows over multiple paths on the basis of the multiplexing character of the multiple flow types.

It is also assumed that the traffic characteristics are stationary or vary slowly as long as the flows are balanced.

The goal is to develop a flow assignment method that assigns flows grouped into $m$ types to $n$ paths on the basis of the traffic characteristics of each group and the number of flows in each group. The criteria for characterizing the flows in order to group them are:

- TCP vs. UDP flows,
- video vs. VoIP streams, where only UDP traffic is focused on by separately treating TCP and UDP traffics (e.g. by assigning different queue classes for them) so that, in some case, the packet loss is minimized for TCP flows and the delay variation is reduced for UDP flows, and

- the bandwidths of the links through which traffic enters the edge router, which connects multiple types of access links, including those aggregating DSL and/or FTTH customers (1-Gbit/s class) and enterprise customers (10-Gbit/s class).

Using these criteria, flows can be groupd into two types (bursty and non-bursty) that have stationary traffic characteristics. Therefore, initially a method for assigning two types of flows is developed. In addition, to avoid unnecessary complexity, the number of paths between each pair of edge routers should be limited to two or three. For simplicity, two-path case is considered for flow assignment.

This method is a first step in developing a method that can easily and efficiently optimize the assignment of actual traffic flows. While a fluid-flow model [Ram99] [DTVV90] or packet-based simulation would enable accurate estimation of queuing delay for each path, their computation takes too long for online use. For real-time application, a method that can quickly estimate the queuing delay for each path so that the flows can be assigned so as to minimize the overall queuing delay is vital.

The estimated flow assignment is used to compute a flow assignment probability which is used for assigning a flow to a path as follows.

1. When a packet arrives, the router checks the routing table and identifies a flow from the incoming packet.
2. The path mapped to the identified flow is retrieved, if the flow had already been active and the mapped path was registered. Otherwise, a path is selected using the flow assignment probability $p_{(l,k)}$ (where $l$ and $k$ denote a path [$l = 1, 2$] and a flow type [$k = 1, 2$], respectively, and $p_{(1,k)} + p_{(2,k)} = 1$), and mapped to the flow. The mapping information is registered for the subsequent packets in the identified flow.

The flow assignment probability is periodically updated in order to adaptively reduce the gap between the actual and estimated number of flows in the paths because the number of actual flows varies over time. The gap is dominated by the dynamics of the flow arrival process, and leads to the actual queuing delay differing from that expected from the estimated flow assignment. There is a trade-off between updating the flow assignment probability more often in order to reduce the gap and conserving the computational resources needed to compute the flow assignment probability. Future improvements to this method include calculating the flow assignment probability on the basis of the flow arrival changes and on the the estimated flow assignment.

### 6.3.2 Delay Metric Representing Delay Performance for Two Paths

In what follows, two different strategies for a good overall delay performance are focused on, whereas there are many possible strategies. One is to minimize the worst (highest) queuing delay among all paths (i.e., the min-max delay strategy) that is based on the max-min fairness concept, and the other is to maximize the sum of logarithm of the difference between a pre-defined queuing delay threshold (an acceptable limit) and the actual queuing delay of individual paths weighted by traffic volume on each path (i.e., the max weighted log-sum delay margin strategy) that may be regarded as a utility function based on application performance.

In correspondence to these strategies, two metrics for overall delay performance over multiple paths termed “min-max delay” and “weighted-log delay” are defined as Equation (6.1) and (6.2), respectively,

$$M_{\min} = \min_{l=1,2}(-d_l(f_{(1,l)}, f_{(2,l)})) \text{ and} \tag{6.1}$$

$$M_{\log} = \sum_{l=1}^{2} w_l \ln\left(\frac{D - d_l(f_{(1,l)}, f_{(2,l)})}{D}\right), \tag{6.2}$$

where $d_l$ denotes one of queuing delay statistics for path $l(l = 1, 2)$ when $f_{(k,l)}$ flows in traffic class $k$ ($k = 1, 2$) are assigned, $w_l$ denotes the ratio of the average rate for path $l$ for the whole traffic and $D$ denotes the threshold of acceptable queuing delay statistics for applications, which may be in the range of 1 ms through 1 s depending on the queuing delay statistics. Factor $w_l$ allows the queuing delay of the path conveying larger amounts of traffic to have a larger weight, which facilitates such a path having a relatively larger queuing delay than the other path.

Maximizing each of these metrics leads to realizing the corresponding strategy.

### 6.3.3 Fluid Flow Model-Based Analysis

The fluid model was first proposed by Anick *et al.* to model data network traffic [AMS82]. In the fluid simulation paradigm, traffic is modeled in terms of a (time) discrete or continuous-rate-based model, rather than discrete packet instances. A fluid simulator technique keeps track of the fluid rate changes at traffic sources and network nodes.

Figure 6.2 shows the model in which an infinite buffer receives two types of on-off state fluid flows. $N_k$ denotes the number of fluid flows of class $k$ and $C$ denotes the fluid output rate. The individual fluid flow of class $k$ comes down at an input rate $r_k$ when it is in the on state, and alternates between the on and off state. The on and off terms are independently and exponentially distributed where the means are $1/\mu_k^{(on)}$ and $1/\mu_k^{(off)}$, respectively. The traffic of UDP flows mentioned above are characterized by the mean of on and off state periods, input rate, and number of fluid flows. The mean rate, $\rho_k$, of a fluid flow in class $k$ and mean rate $\rho$ of the total flows in both classes are represented as $\rho_k = r_k p_k^{(on)}$, and $\rho = \sum_{k=1}^{2} N_k \rho_k$, respectively, where $p_k^{(on)}$ is the steady-state probability in the on state of a class $k$ fluid flow. Here, $\rho < C$, and $\sum_{k=1}^{2} N_k r_k > C$ are assumed. The first assumption makes the fluid flow model stable, and the second ensures the existence of queue length.

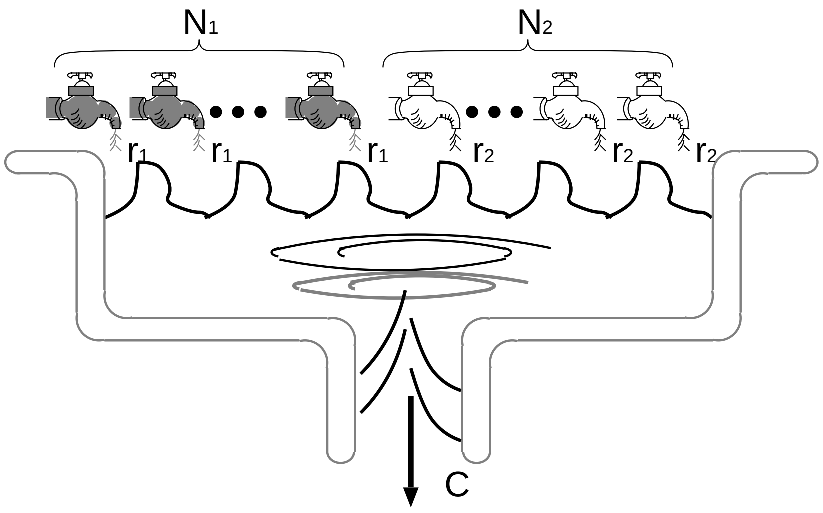


Figure 6.2: Infinite buffer accommodating two types of on-off fluid flow

Let $(i, j) \in \mathcal{S}$ denote the number of on-state fluid flows of classes 1 and 2, $U(t)$ denote the fluid level in the buffer at time $t$, and $c_{(i,j)} = ir_1 + jr_2 - C$ denote the rate of variance of $U(t)$. To simplify the calculations, any $c_{(i,j)}$ is assumed not to be 0. Sets $\mathcal{S}_+$ and $\mathcal{S}_-$ are defined as

$$\mathcal{S}_+ = \{(i, j);\ (i, j) \in \mathcal{S},\ c_{(i,j)} > 0\} \text{ and}$$
$$\mathcal{S}_- = \{(i, j);\ (i, j) \in \mathcal{S},\ c_{(i,j)} < 0\}\ ,$$

where $\mathcal{M}_+, \mathcal{M}_-$, and $\mathcal{M}$ denote the number of states of $\mathcal{S}_+, \mathcal{S}_-$, and $\mathcal{S}$, respectively. Furthermore, the first $\mathcal{M}_+$ states among $\mathcal{S}$ (numbered from 1 to $\mathcal{M}_+$), are included in $\mathcal{S}_+$ and the remaining $\mathcal{M}_-$ states (numbered from $\mathcal{M}_+ + 1$ to $\mathcal{M}$), are included in $\mathcal{S}_-$.

This fluid flow model is an infinite buffer model modulated by the continuous-time Markov chain with $\mathcal{M}$ states, so the transition rate matrix $\mathbf{T}$ and the steady-state probability vector $\boldsymbol{\pi} = (\boldsymbol{\pi}_+, \boldsymbol{\pi}_-) = (\pi_1, \pi_2, \cdots, \pi_{\mathcal{M}})$ satisfy

$$\boldsymbol{\pi}\mathbf{T} = \mathbf{O} \text{ and } \boldsymbol{\pi}\mathbf{e} = 1\ ,$$

where $\mathbf{e}$ is a vertical vector in which all the elements are 1.

To derive the probability distribution $P[\tilde{U} > x]$, and $n$-th moment $E[\tilde{U}^n]$ for the fluid level that the fluid arrivals see and that is denoted by the steady-state random variable $\tilde{U}$, matrices $\mathbf{C}$, $\mathbf{\Pi}$, $\tilde{\mathbf{T}}$, and alignment of sub-matrices including $\mathbf{T}$ are defined as follows,

$$\mathbf{T} = \begin{array}{cc} & \begin{array}{cc} \mathcal{M}_+ & \mathcal{M}_- \end{array} \\ \begin{array}{c} \mathcal{M}_+ \\ \mathcal{M}_- \end{array} & \begin{pmatrix} \mathbf{T}_{+,+} & \mathbf{T}_{+,-} \\ \mathbf{T}_{-,+} & \mathbf{T}_{-,-} \end{pmatrix} \end{array}, \quad \mathbf{C} = \begin{array}{cc} & \begin{array}{cc} \mathcal{M}_+ & \mathcal{M}_- \end{array} \\ \begin{array}{c} \mathcal{M}_+ \\ \mathcal{M}_- \end{array} & \begin{pmatrix} \mathbf{C}_+ & \mathbf{O} \\ \mathbf{O} & \mathbf{C}_- \end{pmatrix} \end{array}, \quad \mathbf{\Pi} = \begin{array}{cc} & \begin{array}{cc} \mathcal{M}_+ & \mathcal{M}_- \end{array} \\ \begin{array}{c} \mathcal{M}_+ \\ \mathcal{M}_- \end{array} & \begin{pmatrix} \mathbf{\Pi}_+ & \mathbf{O} \\ \mathbf{O} & \mathbf{\Pi}_- \end{pmatrix} \end{array},$$

$$\text{and } \tilde{\mathbf{T}} = \begin{array}{cc} & \begin{array}{cc} \mathcal{M}_+ & \mathcal{M}_- \end{array} \\ \begin{array}{c} \mathcal{M}_+ \\ \mathcal{M}_- \end{array} & \begin{pmatrix} \tilde{\mathbf{T}}_{+,+} & \tilde{\mathbf{T}}_{+,-} \\ \tilde{\mathbf{T}}_{-,+} & \tilde{\mathbf{T}}_{-,-} \end{pmatrix} \end{array} = \mathbf{\Pi}^{-1}\mathbf{T}^T\mathbf{\Pi},$$

where $\mathbf{A}^T$ is the transpose of $\mathbf{A}$. The probability distribution and $n$-th moment for the fluid level are derived, as follows [Ram99],

$$P[\tilde{U} > x] = \frac{\alpha \exp(x\mathbf{V})\mathbf{r}_+}{\pi\mathbf{r}} \text{ and} \tag{6.3}$$

$$E[\tilde{U}^n] = \frac{n!\alpha\{(-\mathbf{V})^{-1}\}^n\mathbf{r}_+}{\pi\mathbf{r}}, \tag{6.4}$$

respectively, where $\mathbf{V}$ is a $\mathcal{M}_+ \times \mathcal{M}_+$ matrix in which all the diagonal elements are negative and non-diagonal elements are positive, $\alpha = \boldsymbol{\pi}_+ + \boldsymbol{\pi}_-\mathbf{W}$, and $\mathbf{r} = (\mathbf{r}_+, \mathbf{r}_-) = (r_1, r_2, \cdots, r_{\mathcal{M}})$ is a vertical vector in which an element is the total input rate of each state $m$ ($m \in \{1, 2, \ldots, \mathcal{M}\}$). $\mathbf{V}$ and $\mathbf{W}$ satisfy the following

$$\mathbf{V} = \mathbf{C}_+^{-1}\tilde{\mathbf{T}}_{+,+} + \mathbf{C}_+^{-1}\tilde{\mathbf{T}}_{+,-}\int_0^\infty \exp\{y(-\mathbf{C}_-)^{-1}\tilde{\mathbf{T}}_{-,-}\}(-\mathbf{C}_-)^{-1}\tilde{\mathbf{T}}_{-,+}\exp(y\mathbf{V})dy$$

$$\mathbf{W} = \int_0^\infty \exp\{y(-\mathbf{C}_-)^{-1}\tilde{\mathbf{T}}_{-,-}\}(-\mathbf{C}_-)^{-1}\tilde{\mathbf{T}}_{-,+}\exp(y\mathbf{V})dy\ .$$

From Equation (6.3) and (6.4), given a set of $f_{(k,l)}$ flows of class $k$ ($k = 1, 2$) in path $l$ ($l = 1, 2$), the mean $d_{mean}$, standard deviation $d_{stdev}$, and 99.5th percentile $d_{99.5^{\text{th}}}$ of queuing

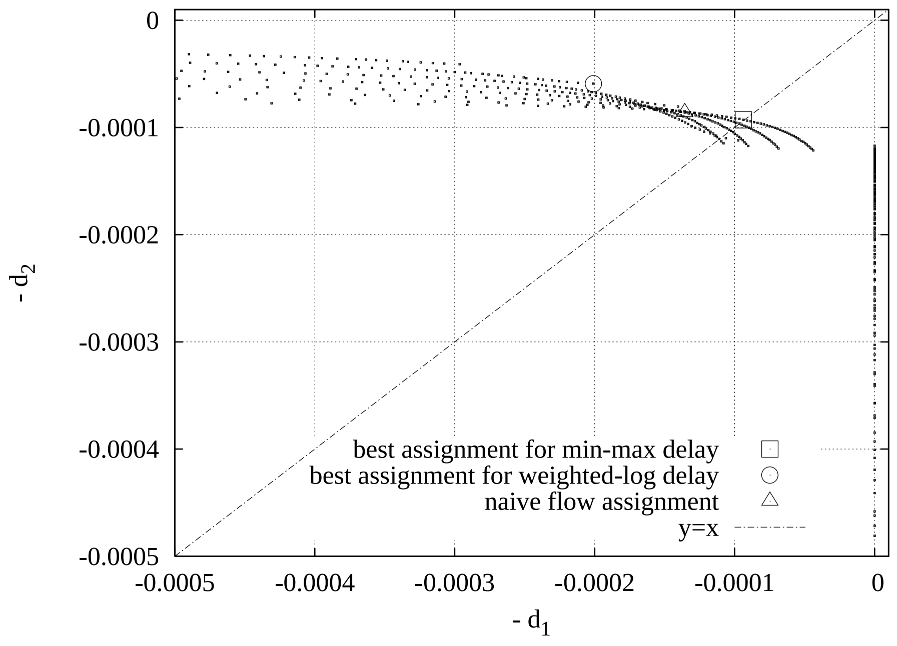


Figure 6.3: The mean of queuing delay for two path with the total utilization 0.6

delay are derived as follows:

$$
\begin{aligned}
d_{mean}(f_{(1,l)}, f_{(2,l)}) &= \frac{E[\tilde{U}_l]}{C_l}, \\
d_{stdev}(f_{(1,l)}, f_{(2,l)}) &= \frac{\sqrt{E[\tilde{U}_l^2]-E[\tilde{U}_l]^2}}{C_l} \text{ , and} \\
d_{99.5}\text{th}(f_{(1,l)}, f_{(2,l)}) &= \frac{x_l}{C_l} \text{ ,}
\end{aligned}
\tag{6.5}
$$

respectively, where $C_l$ denotes the bandwidth of path $l$, and queue length $x_l$ satisfies $P[\tilde{U}_l > x_l] = 1 - 0.995$ .

### 6.3.4 Relationship between Delay Metric and Best Flow Assignment

Figure 6.3 shows the mean of the queuing delay for possible flow assignments to two paths with bandwidths of 4 and 8 Mbit/s. The characteristics of the traffic flows is same as the the traffic combination 1 in Table 6.2 described in Section 6.4. Two paths are shared by 7 high-rate flows and 174 low-rate flows. The average utilization of total path capacity is 0.6. The acceptable delay threshold ($D$) is defined as 0.01 s, and $d_1$ and $d_2$ as the mean of queuing delays for the 4-Mbit/s and 8-Mbit/s paths, respectively.

With respect to the min-max delay, the delay metric for a flow assignment is represented by the value on the X-axis if its point are above the line $y = x$, and by the value on the Y-axis, otherwise. The highest delay metric is delivered by the flow assignment (indicated by $\square$): 92 low-rate flows to the 4-Mbit/s path and the rest to the 8-Mbit/s path, where the queuing delays are 0.0000936 s and 0.0000928 s, respectively. With respect to the weighted-log delay, the highest delay metric is delivered by the flow assignment (indicated by $\bigcirc$): 6 high-rate flows to the 4-Mbit/s path and the rest to the 8-Mbit/s path, where the queuing delays are 0.000201 s and 0.0000591 s, respectively. The mark $\triangle$ represents the *naive flow assignment* which means that the individual traffic classes are divided in proportion to the ratio of path bandwidths, where 2 high-rate and 58 low-rate flows are assigned to the 4-Mbit/s path, and the rest to the 8-Mbit/s path. The best flow assignment for the min-max delay equalizes the queuing delays between paths. However, for the weighted-log delay, the best flow assignment has a larger gap between the queuing delays in individual paths. The points falling onto the line $x = 0$ mean that there is no queuing delay in the 4-Mbit/s path because the peak rate of the whole traffic assigned to this path is less than the path bandwidth (4 Mbit/s). This signifies that the flow assignment is not adequately balanced between two paths in terms of the queuing delay.

Figure 6.4 shows another case where the metric exploits the 99.5$^{\text{th}}$ percentile of the queuing delay, the acceptable delay threshold is 0.1 s, and the flows are composed of 21 high-rate and 177 low-rate flows so that the utilization of the total path capacity is 0.9. With respect to the min-max delay, the best flow assignment ($\square$) is 126 low-rate flows assigned to the 4-Mbit/s path and the rest to the 8-Mbit/s path, where the queuing delays in the 4- and 8-Mbit/s paths are 0.01697 s and 0.01602 s, respectively. In the case of the weighted-log delay,

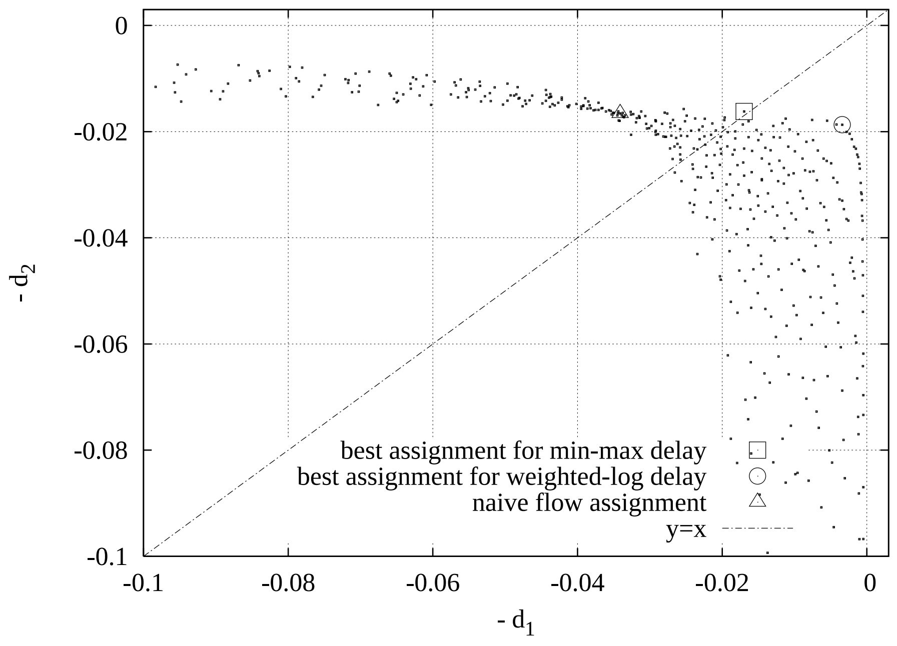


Figure 6.4: The 99.5th percentile of queuing delay for two path with the total utilization 0.9

the best flow assignment (◯) is achieved by 121 low-rate flows assigned to the 4-Mbit/s path, and the rest to the 8-Mbit/s path, where the queuing delays of the 4- and 8-Mbit/s paths are 0.00342 s and 0.01871 s, respectively.

Comparing the min-max and weighted-log delays, although the queuing delay of the 8-Mbit/s path is slightly worse for the weighted-log delay than for the min-max delay, the delay of the 4-Mbit/s path is adequately improved in compensation. In addition, Comparing the 0.6 and 0.9 utilization, the property of the best flow assignments for the weighted-log delay are completely different: the former has larger queuing delay in the 4-Mbit/s path while the latter has in the 8-Mbit/s path. This signifies that it is not trivial to find the best flow assignment under the weighted-log delay. These tendencies can be seen for the other queuing delay statistics (the mean and standard deviation), traffic characteristics, and different path bandwidths.

Table 6.1: Path combinations over which traffic flows are balanced

| Path combination | 1 | 2 |
|---|---|---|
| Path 1 | 6 Mbit/s | 4 Mbit/s |
| Path 2 | 6 Mbit/s | 8 Mbit/s |

Table 6.2: Traffic characteristics and combinations of two traffic classes

| Traffic combination | 1 | | 2 | |
|---|---|---|---|---|
| Traffic class | H-1 | L-1 | H-2 | L-2 |
| Average period of on-state | 10 ms | 10 ms | 20 ms | 5 ms |
| Average period of off-state | 30 ms | 10 ms | 20 ms | 15 ms |
| Probability of on-state | 0.25 | 0.5 | 0.5 | 0.25 |
| Rate in on-state | 1 Mbit/s | 64 Kbit/s | 512 Kbit/s | 128 Kbit/s |
| Average rate of a flow | 256 Kbit/s | 32 Kbit/s | 256 Kbit/s | 32 Kbit/s |

## 6.4 Effectiveness of Traffic Characteristics-aware Flow Assignment

The the mean, standard deviation, and 99.5th percentile of queuing delay for the possible flow assignment to two paths with respect to the various numbers of flows and the different combinations of path bandwidths are analyzed. In the analysis two combinations for the path as showed in Table 6.1 were employed.

Combination 1 and 2 consist of the same and different bandwidths of paths, respectively. Two traffic combinations, as showed in Table 6.2. are also employed. H-1 and H-2 flows are termed the *high-rate flows* and L-1 and L-2 flows are termed the *low-rate flows*, hereafter.

In the preliminary experiments the comparison of the flow assignments making the delay

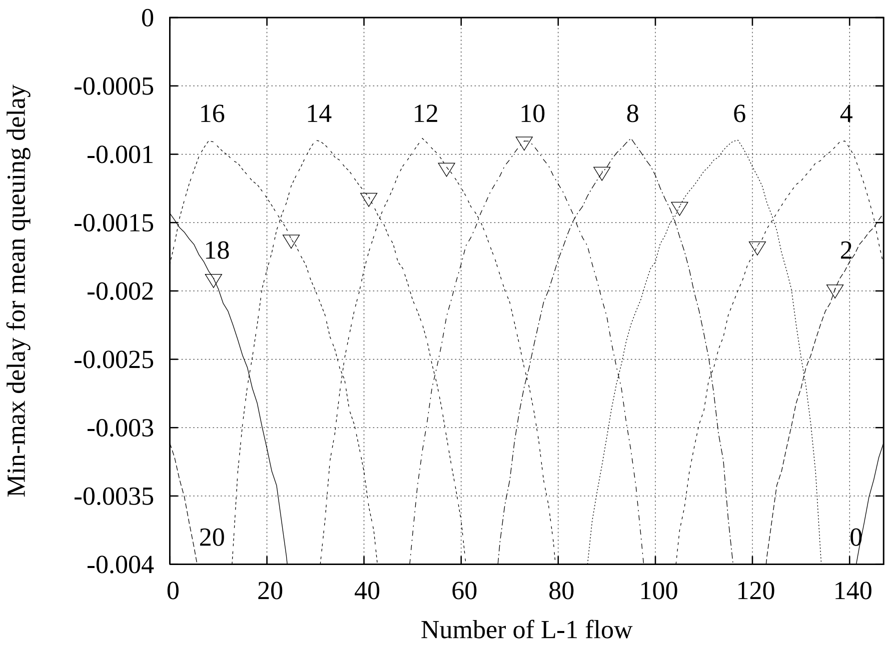


Figure 6.5: Min-max delay for mean queuing delay for traffic combination 1 distributed over two 6-Mbit/s paths

metrics high with respect to individual queuing delay statistics, showed that the number of flows for the flow assignments were very similar each other and that the difference was few number in low-rate flows. Such a feature was showed for any pair of the traffic combination and path combination. Therefore, the mean of queuing delay is employed for the analysis in the following subsections.

### 6.4.1 Best Flow Assignment under Same-bandwidth Paths

First, the best flow assignment for a sufficient number of flows with same-bandwidth paths (path combination 1) is analyzed. The numbers of high-rate and low-rate flows are 20 and 147, respectively, for traffic combination 1 and 2. The average rates for both traffic combinations are 5.12 Mbit/s for high-rate flows and 4.70 Mbit/s for low-rate flows, and $\rho$ is 9.82

Mbit/s in total, so that the total utilization for two paths is about 80 %. The min-max delay for traffic combination 1 is showed in Figure 6.5. The number near a line and the X-axis indicate the number of high-rate and low-rate flows, respectively, that are assigned to the path 1, i.e., a points on a line labeled ’12’ at 60 on the X-axis indicates that 12 high-rate flows and 60 low-rate flows are assigned to path 1 and the 8 high-rate flows and 87 low-rate flows to path 2.

From the figure, there are multiple flow assignments making the delay metric high. The naive flow assignment, that is, the assignment equally dividing flows in both traffic classes also makes the delay metric high. The $\triangledown$ marks in the figure indicate combinations of the number of flows that lead to equal utilization between paths. $\triangledown$ points incur the lower delay metric as the difference in the number of high-rate and low-rate flows between paths becomes greater. The delay metrics can be increased by shifting the number of low-rate flows from such an assignment so as to make the utilization excessively unbalanced between paths. These tendencies: the existence of multiple high-delay-metric assignments and shifting low-rate flows from equal utilization assignments, were also found in traffic combination 2.

In terms of network operation, the naive flow assignment in both traffic classes is much simple and, therefore, desirable. On the other hand, the existence of multiple flow assignments lowering the delay is also promising to apply the flow assignment in the case where either or both traffic classes cannot be equally divided. For example, flows in a traffic class require a short one-way delay, and only one path can satisfy the requirement.

The weighted-log delay for traffic combination 2 with path combination 1 is showed in Figure 6.6. The traffic characteristics and number of flows are same as those of the min-max delay in Figure 6.5. The best flow assignment is: 18 high-rate flows are in one path and the rest are in the other path. In contrast to the min-max delay, the all the equal-utilization flow assignments ($\triangledown$ marks) make the delay metric high besides the naive flow assignment. The

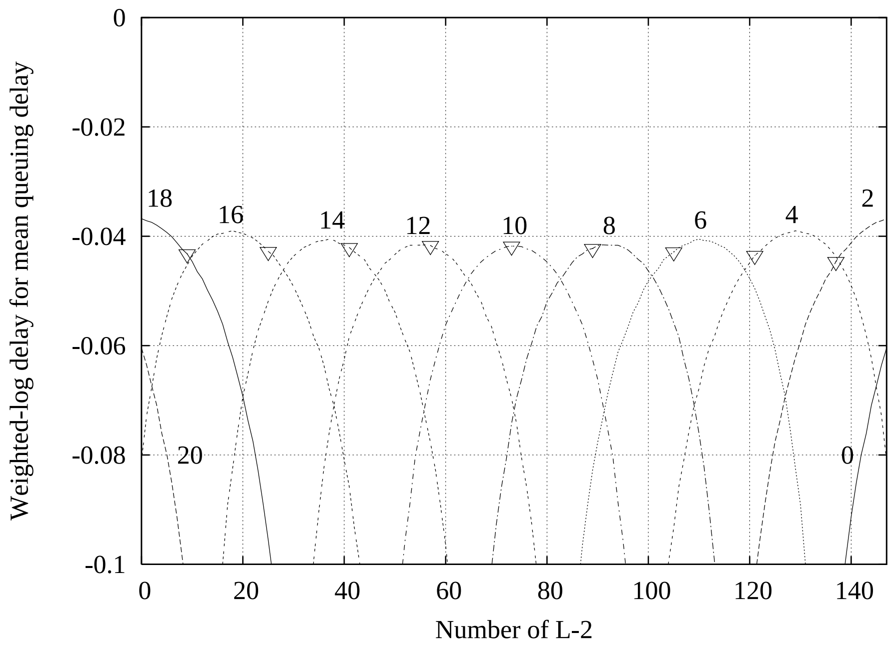


Figure 6.6: Weighted-log delay for mean queuing delay for traffic combination 2 distributed over two 6-Mbit/s paths

delay metric is increased by shifting the number of low-rate flows from such a assignment so as to make the utilization excessively unbalanced between paths.

These tendencies: the existence of multiple high-delay-metric assignments, similarly high delay metrics among all the equal-utilization assignments, and shifting low-rate flows from equal utilization assignment, were also found in traffic combination 1.

## 6.4.2 Best Flow Assignment under Different-bandwidth Paths

The best flow assignment for the min-max delay in distributing flows over different-bandwidth paths, path combination 2 is analyzed.

The number of flows is the same as that of path combination 1. The delay metric is showed in Figure 6.7. The number near a line and X-axis also indicate the number of high-

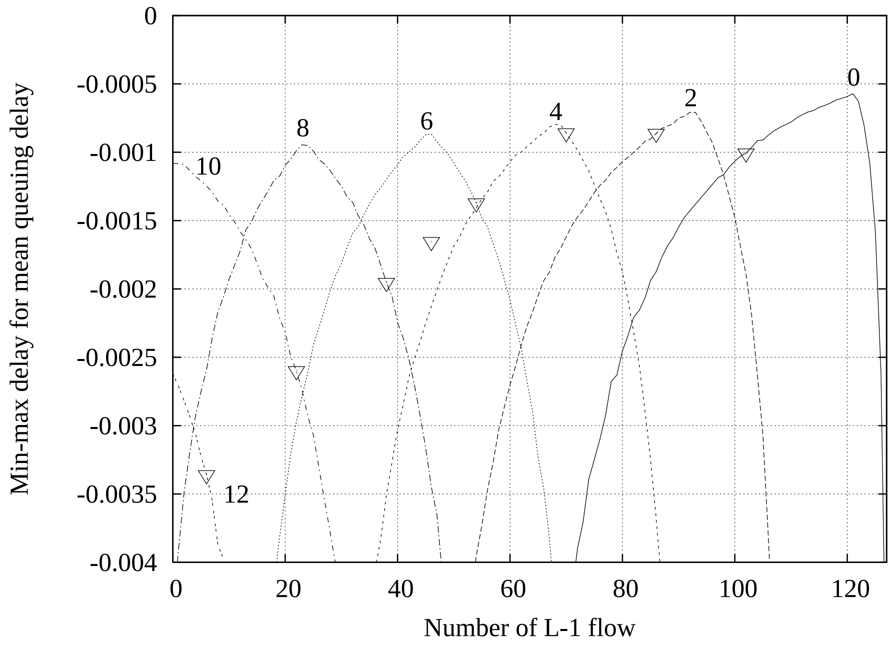


Figure 6.7: Min-max delay for the mean queuing delay for traffic combination 1 distributed over 4-Mbit/s and 8-Mbit/s paths

rate and low-rate flows, respectively, assigned to path 1: the 4-Mbit/s path.

The figure indicates that the naive flow assignment, that is, 7 high-rate flows and 46 low-rate flows assigned to the 4-Mbit/s path and the rest to 8-Mbit/s where the delay metric is -0.00165, cannot maximize the delay metric. Some of equal-utilization assignment can make the delay metric higher than the naive flow assignment does. The flow assignment making the utilization between two paths unbalanced excessively by shifting low-rate flows from the equal-utilization assignment (▽) is as effective as path combination 1 for making the delay metric higher. These tendencies were also found in traffic combination 2.

The assignment: all the high-rate flows are assigned to the 8-Mbit/s path minimizes the delay metric. Such an assignment is probably because there are sufficient low-rate flows. However, it is not trivial to find the exact number of low-rate flows or the range of number

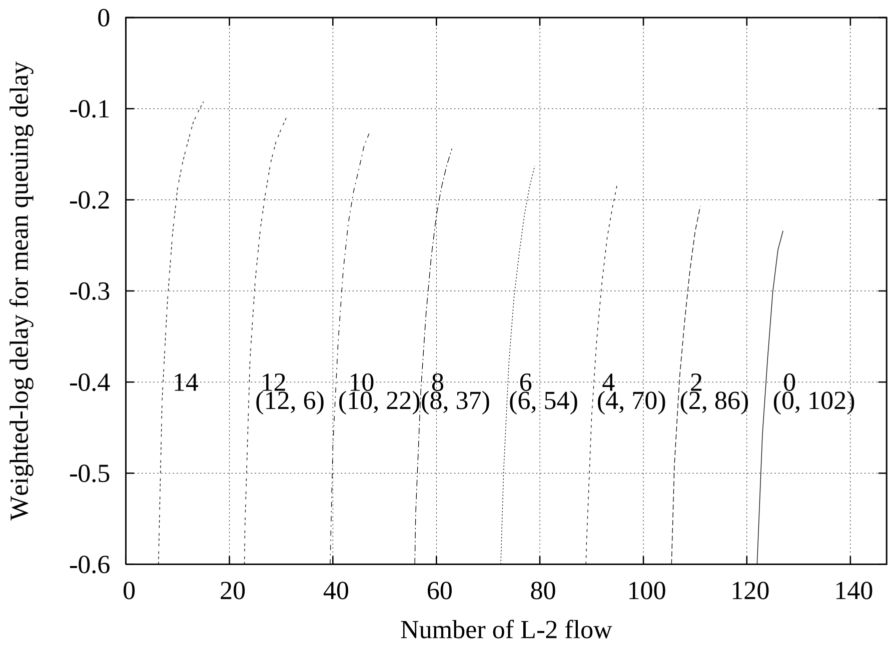


Figure 6.8: Weighted-log delay for mean queuing delay for traffic combination 2 distributed over 4-Mbit/s and 8-Mbit/s paths

to divide the low-rate flows so as to make the delay metric high. Actually, with respect to traffic combination 2, the best flow assignment: 112 low-rate flows assigned to the 4-Mbit/s the rest to the 8-Mbit/s differed from traffic combination 1.

Figure 6.8 shows the weighted-log delay for traffic combination 2 with different-bandwidth paths, path combination 2. The discontinuity of line is because the average path utilization of the 4-Mbit/s path reaches to or exceeds 1. A pair of numbers expressed by $(H, L)$ near a line shows the numbers of flows denoting high-rate flows by $H$ and low-rate flows by $L$, which make the path utilization equal between two paths. The figure shows that any equal-utilization assignment makes the delay metric significantly low.

To find features such that flow assignment maximizing the delay metric over the different-bandwidth paths, the other combinations of the number of flows with path combination 2

Table 6.3: Number of flows classified based on peak rate

| Traffic class | Number of flows | | |
|---|---|---|---|
| | Small | Middle | Large |
| H-1 | 3 | 6 | 12 |
| H-2 | 6 | 12 | 24 |
| L-1 | 50 | 100 | 200 |
| L-2 | 25 | 50 | 100 |

were analyzed. This was carried out by classifying the peak rates $r_k f_{(k,l)}$ of aggregated flows. The labels, 'small,' 'middle,' and 'large' in Table 6.3 represent whether the peak rate of an individual traffic class is less than the 4-Mbit/s path, between the 4-Mbit/s and 8-Mbit/s paths, or more than the 8-Mbit/s path, respectively. In all combinations of the number of flows from H-1 and L-1, or H-2 and L-2, there are six combinations incurring the queuing delay in each traffic class combination. Instances of flow assignment maximizing the min-max delay with respect to those combinations are presented in Table 6.4. The labels 'narrow', 'broad', and 'both' expressed as $(A, B)$ in the columns indicate that all the flows are assigned to the 4-Mbit/s path, the 8-Mbit/s path, and divided into both paths, respectively, with the high-rate flows denoted by $A$ and the low-rate flows denoted by $B$.

The table indicates that there are simple flow assignment, that is, high-rate and low-rate flows assigned to the 8-Mbit/s and 4-Mbit/s path, respectively, for the peak rate combinations: (middle, middle) and (large, small) in traffic combination 1. However, with respect to traffic combination 2, both high-rate and low-rate flows should be divided into both paths for the same peak rate combinations. In addition, the instances of flow assignment for individual peak rate combinations differ depending on the traffic combinations. This tendency was also showed in the case of the weighted-log delay. These results suggest that taking into account

Table 6.4: Flow assignment maximizing the min-max delay for each number of flows on individual traffic combinations

| Number of flows | | How to divide | |
|---|---|---|---|
| H-1 or H-2 | L-1 or L-2 | Assignment of H-1 and L-1 | Assignment of H-2 and L-2 |
| Large | Large | (broad, both) | (both, both) |
| Large | Middle | (broad, both) | (broad, both) |
| Middle | Large | (broad, both) | (both, both) |
| Middle | Middle | (broad, narrow) | (both, both) |
| Large | Small | (broad, narrow) | (both, both) |
| Small | Large | (broad, both) | (both, both) |

the traffic characteristics of traffic flows is a key to find the best flow assignment in terms of the delay performance.

### 6.4.3 Summary

Through the numerical analysis, It was found that: 1) the high-delay-metric flow assignments obtained with one of the queuing delay statistics are very similar to those obtained with the other queuing delay statistics; 2) dividing flows in both traffic classes in proportion to the ratio of path bandwidth (naive assignment) can make the delay metrics high when two paths have the identical bandwidths; and 3) there are several assignments with better delay metrics than the naive assignment in the case where the path bandwidths are different. Therefore, in the following section, the case of different path bandwidths is focused and a method for finding an alternate flow assignment, that makes the delay metrics higher than the naive flow assignment does, is proposed. It is showed that the delay performance obtained by proposed method is remarkably better than that by the naive flow assignment and that the approach works with both delay metrics.

## 6.5 Flow Assignment Method

As mentioned, the proposed flow assignment method reduces the queuing delay by taking into account the traffic characteristics of each flow types. It is targeted at the case where the path bandwidths are different as described in Section 6.4.3. When a set of flows are given, the method takes two steps for finding the best flow assignment

1. approximately placing delay contour lines, as described in Section 6.5.1, and
2. searching the best (highest min metric) flow assignment by iteratively estimating the queuing delay for a certain number for flows from a given set of flows are assigned to one of two paths with the delay contour lines.

The contribution of this study is an algorithm for the first step which places the delay contour lines on the basis of the path bandwidth and traffic characteristics of two flow types. The obtained delay contour lines allow the second step to approximately estimate the queuing delay promptly.

For the second step, the hill-climbing algorithm is applied to the search for the best flow assignment. In the search process, the algorithm iterates the following steps until the highest min metric is found

1. dividing flows into two paths,
2. estimating the queuing delay for each path with the delay contour lines, and

3. computing the min metric from the two queuing delays.

In every instance of the iteration, the flows are divided into the different numbers.

In the following subsections, firt, the queuing delay characteristics for a single path is described in order to indicate the nature of the delay contour lines in Section 6.5.1. Second, the algorithm approximately determining the placement of the delay contour lines is described in Section 6.5.2. And lastly, the hill-climbing algorithm for searching for the best flow assignment using the delay contour lines is described in Section 6.5.3.

### 6.5.1 Queuing Delay Characteristics

First, class 1 and class 2 flows are defined with a rule: the steady state probability of the on-state for a class 1 flow, denoted by $p_1$, is less than or equal to that for a class 2 flow, denoted by $p_2$ ($p_1 \leq p_2$). Given a set of flows to be assigned to a path, the numbers of each flows are transformed to the delay function $\tilde{d}(u, r)$ parameterized by the average path utilization $u$ ($0 \leq u < 1$) and the average ratio of class 1 flows $r$ ($0 \leq r \leq 1$) to total flows.

Figure 6.9 shows the relationship between the average path utilization (x-axis) and the ratio of class 1 flows (y-axis) for a single path. The flows have the same traffic characteristics as described in Section 6.3.4, and the path bandwidth is 4 Mbit/s. All the points on each line have the same 99.5$^{\text{th}}$ percentile queuing delay. Points on the x-axis (or $r = 1$), where the ratio of class 1 flows is 0 (1), represent the path occupied by only the class 2 (1) flows. This delay curve is called the 'delay contour line' hereafter.

From observations of various combinations of flow characteristics and path bandwidths, the followings are found:

- the delay contour lines do not intersect;
- the larger the utilization, the larger the queuing delay; and

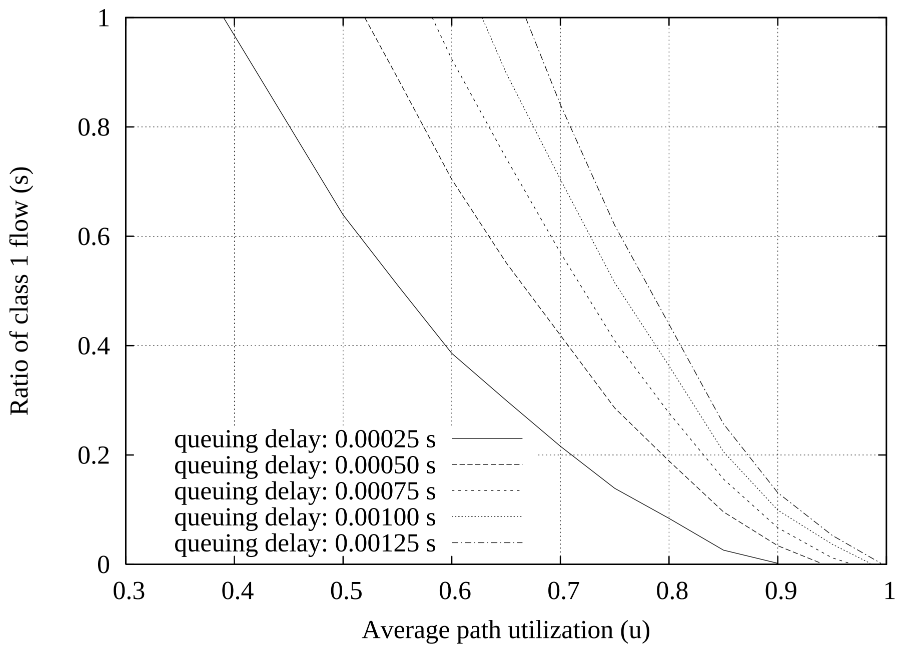


Figure 6.9: Delay contour line for $99.5^{mboxth}$ percentile queuing delay against ratio of class-1 flows and average path utilization for a single path.

- there exist multiple points (flow assignments) having the same queuing delay.

The figure shows that reducing either the ratio of class 1 flows or the average path utilization reduces the queuing delay when flows are assigned to a single path. The nature of the delay contour lines was the same as the the other queuing delay statistics (the mean queuing delay and the standard deviation of queuing delay) although the gradient of the delay contour lines and the space between the lines differed depending on the path bandwidth and/or the combination of traffic characteristics.

For a given set of flows, the proposed search algorithm (for the second step) can easily estimate the queuing delay from the nearest delay contour lines once the lines for each path have been suitably placed using the algorithm described in the following section.

### 6.5.2 Algorithm for Approximately Placing Delay Contour Lines

In considered flow assignment approach (described in Section 6.3.1), it is assumed that the number of flows varies between assignment periods. The used queuing delay statistic, such as, the mean queuing delay, is acquired every period. This means that the three-tuple data (average path utilization, ratio of class 1 flows, and queuing delay) are collected every period. the proposed algorithm for placing the delay contour lines uses these data and the amount of data increases as the assignment period proceeds. Note that initially, the flows are assigned so as to maintain equal path utilization (the naive flow assignment). Once sufficient three-tuple data has been collected, the flows are assigned using the proposed method.

The algorithm first identifies an appropriate area for placing the delay contour lines. Given a set of flows to be assigned to a path, the sum of the peak rate (denoted by $R_k$) of all the flows must exceed the path bandwidth (denoted by $c_l$) so that there is a queuing delay $\tilde{d}(u, r) > 0$,

$$R_1 f_{(1,l)} + R_2 f_{(2,l)} > c_l \ . \tag{6.6}$$

If the average rate of each flow type are given by $p_1 R_1 f_{(1,l)} = ruc_l$ and $p_2 R_2 f_{(2,l)} = (r-1)uc_l$, $u$ and $r$ hold

$$\begin{array}{lll} r > \frac{p_1}{p_2 - p_1}\left(\frac{p_2}{u} - 1\right) & \text{if} \ \ p_1 < p_2 \\ u > p_1 & \text{otherwise } (p_1 = p_2), \end{array} \tag{6.7}$$

which specify where the delay contour lines should be placed.

It is hypothesized that the delay contour lines have the form,

$$r = a/u + b \quad (a > 0), \tag{6.8}$$

from Inequality (6.7). If this hypothesis holds, they can approximately placed by deriving parameters $a$ and $b$ when two three-tuple data having identical queuing delays are given. The ease of placing the delay contour lines is a key feature of the proposed algorithm. Preliminary experiments using various parameters for the traffic characteristics and the various combinations of path bandwidths, showed that The delay contour lines derived by Equation (6.8) with two three-tuple data nicely fitted in the result of the numerical analysis with fluid-flow model. Therefore, Equation (6.8) is used for approximately placing the delay contour lines.

To approximately place the lines using a single three-tuple data, simultaneously estimating the placement of several of them in an iterative manner under two restrictions.

r-1 Equation (6.8) satisfies $u < 1$ and Inequalities (6.7) for $0 \leq r \leq 1$.

r-2 The delay contour lines do not intersect, as mentioned in Section 6.5.1.

The delay contour lines ($j = 1, 2, \cdots$) are numbered in the order of queuing delay, denoted by $q_j$, where $q_j < q_{j+1}$. A delay contour line $j$ and the amount of $a_{(j,i)}$ being moved for the line $j$ are defined as $r = a_{(j,i)}/u + b_{(j,i)}$, and $\alpha_j$, respectively. Note that $b_j$ is derived from $a_j$ and the given three-tuple data in each iteration. A pair of parameters $a_{(j,i)}$ and $b_{(j,i)}$ moves the delay contour lines independently.

The algorithm works as follow.

1. The initial values are set to parameters $a_{(j,0)}$ and $\alpha_j$. An example of $a_{(j,0)}$ is $\frac{p_1 p_2}{p_2 - p_1}$ if $p_1 < p_2$.
2. In iteration $i$, $a_{(j,i)}$ is updated by increasing or decreasing $a_{(j,i-1)}$ by $\alpha_j$ on the basis of the following rules.
   (a) If delay contour line $j$ violates r-1 $a_{(j,i)}$ is increased.
   (b) if delay contour lines $j$ and $j + 1$ intersect (violate r-2), $a_{(j,i)}$ is increased (decreased) and $a_{(j+1,i)}$ is decreased (increased) if intersection point is lower (higher) than the three-tuple data for each line.

The lines are moved starting with the one with the largest queuing delays because the a higher delay contour line is more likely to violate r-1. $\alpha_j$ is halved if $a_j$ continues to move back and forth.

3. The iteration is terminated when all the lines satisfy r-1 and r-2 or when $\alpha_j$ is less than $\epsilon$, which is a sufficiently small value.

If there are delay contour lines representing multiple three-tuple data, the algorithm retains their positions and moves the others.

A preliminary experiment showed that the computational cost of this algorithm was low enough for it to be used in an online manner for a moderate number of delay contour lines (10 – 50).

### 6.5.3 Search for Best Flow Assignment

During the search for the best flow assignment using the hill-climbing algorithm, most of the possible flow assignments do not fall exactly onto a delay contour line. Given a flow assignment denoted by $v(u_x, r_x)$, the algorithm first identifies the two delay contour lines closest to $v$. Next, the queuing delay is linearly derived from the queuing delays represented by those two lines. In this process, the distance is measured with respect to the horizontal line from $v$ ($r = r_x$). Although using four or more delay contour lines would enable more accurate estimation of the queuing delay, only two are used in the proposed method, because even though the delay contour line placement includes a more inaccurate factor, the estimated flow assignment still results in a good queuing delay, as described in Section 6.5.4.

The way of dividing a given set of flows into two paths in the search process is:

1. initially the number of naive flow assignment is used;

2. the algorithm shifts a small amount flows of one flow type (flow type $A$) from one path to the other from the flow assignment in the previous iteration until the min metric reaches a local maximum; and

3. Next, the algorithm shifts a small amount of flows of the other flow type (flow type $B$) and returned to the movement of the flow type $A$. until the local maximum for step 2 no longer increases.

The amount of shifted flows should reflect the amount of the increase of min metric (the larger increasing, the larger amount of flows are shifted).

### 6.5.4 Evaluation

The delay metrics obtained from the proposed method are compared with that from the naive flow assignment. Figures 6.10 and 6.11 show delay metrics of the proposed method normalized with respect to the naive flow assignment for the 99.5th percentile of queuing delay in the cases of the min-max delay (Figure 6.10) and weighted-log delay (Figure 6.11). For those figures, path combination 2 and traffic combination 1, as described in Section 6.4, were employed. The X-axis represents the average utilization for two paths. The Y-axis represents the average ratio of a hundred examinations. A ratio lower than one means that the delay metrics obtained by the proposed method, were higher (better) than those obtained by the naive flow assignment because the value of metric is negative. The legend shows the number of three-tuple data used to produce the delay contour lines. In each of the examination, all the possible flow assignments were examined for each of the average utilization with randomly selected three-tuple data.

For the min-max delay, the results show that the proposed method outperformed the naive flow assignment for all the utilization. The decrease in the delay metric ranged from 15 through 40 %, which is a remarkable improvement. Similar improvements were seen for

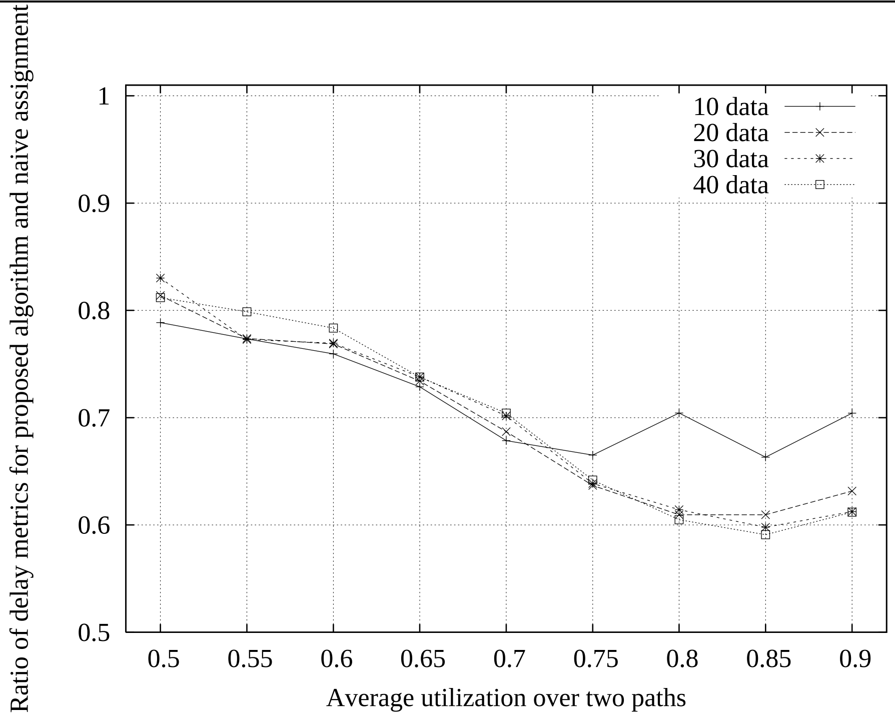


Figure 6.10: Improvement under the min-max delay strategy

the cases of 20 through 40 sample data. For the weighted-log delay, the proposed method outperformed by 10 – 15 % the naive flow assignment under higher utilization from 0.7 to 0.9, although efforts to make the delay metrics high failed with few data (10) under low utilization from 0.5 to 0.65. These tendencies were found in cases with other queuing delay statistics (mean and standard deviation), different combinations of path bandwidths, and distinct traffic characteristics.

Figures 6.12 and 6.13 shows the queuing delay of the best flow assignment obtained from hundred examinations, for the min-max strategy and the max weighted log-sum delay margin strategy, respectively, when only 20 sample data were used in the case of 80 % utilization. Figure 6.13 has the converted axises for easily distinguishing improvement of the flow assignment. Here, the good flow assignment should have the high Y-axis interception of the line with the gradient -1 onto which the flow assignment point falls. The estimations

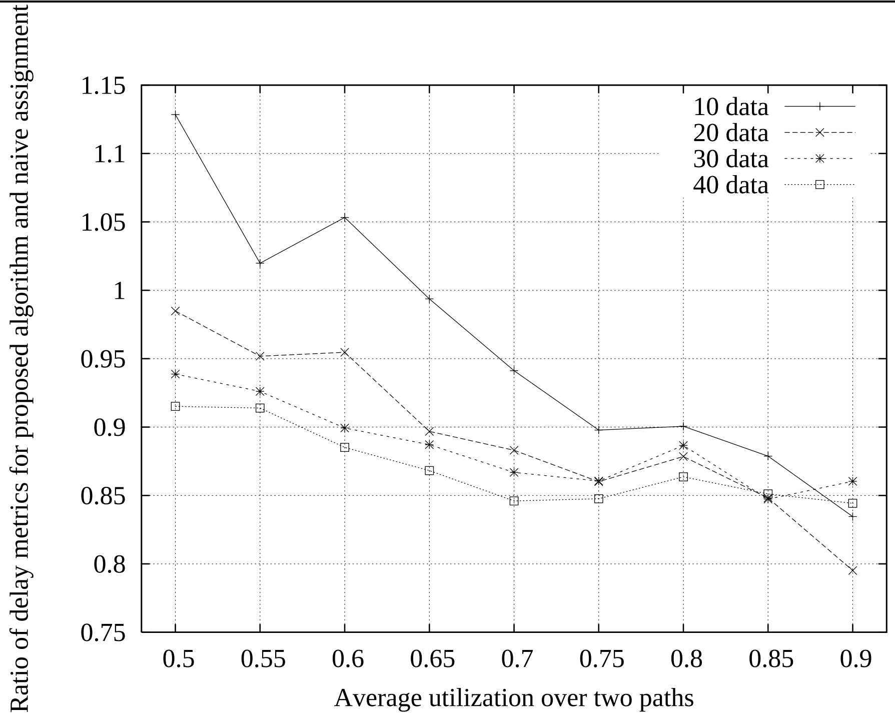


Figure 6.11: Improvement under the max weighted log-sum delay margin strategy

for some examinations gave the same flow assignments for both figures. With respect to the min-max delay (Figure 6.12), every estimation gives a larger delay metric than the naive flow assignment, which probably explains the remarkable improvement.

The smaller improvement of the max weighted log-sum delay margin strategy (Figure 6.13) than min-max delay strategy is probably because some estimations gave the delay metric similar to or higher than those of naive flow assignment.

## 6.6 Conclusion

In this work, It has been considered how a number of flows having distinct traffic characteristics should be distributed over multiple network paths to achieve a good overall delay performance. Two delay metrics were defined as the delay performance metrics representing

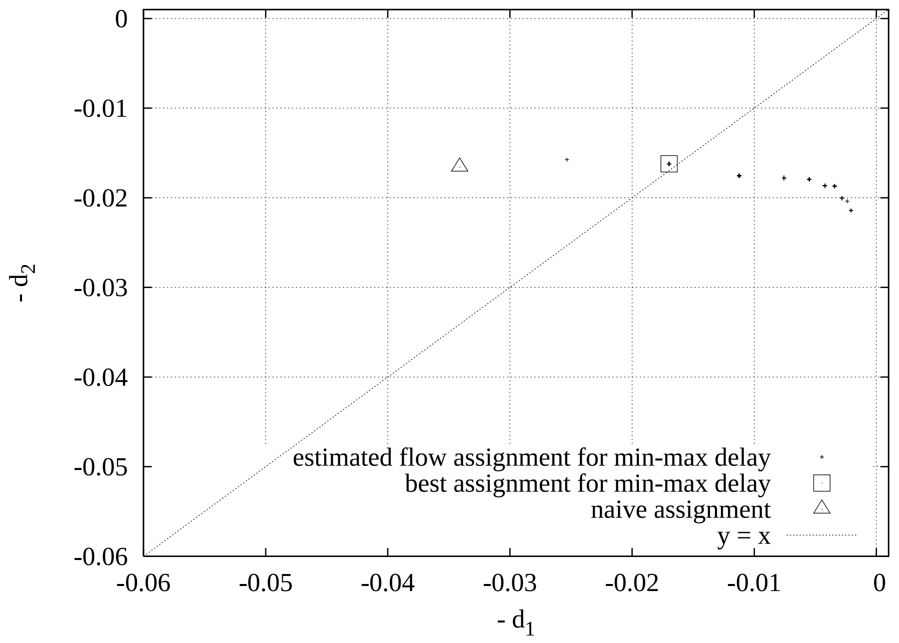


Figure 6.12: The 99.5th percentile of queuing delay for estimated flow assignment with min-max delay for the case of only 20 sample data and the 80 % utilization

the queuing delay for two paths: the min-max delay and weighted-log delay. Then, using a fluid-flow model-based numerical analysis, it was showed that a naive flow assignment based on the conventional path bandwidth-based flow assignment could not maximize these delay metrics when the flows were distributed over different-bandwidth paths, and that the optimal flow assignment would be non-trivial in this context.

Next, an algorithm, which could estimate the queuing delays for an arbitrary flow assignment from the measured queuing delays for a set of sample flow assignments previously observed, was developed. This algorithm can work without an exact analytical formula on the relationship between traffic characteristics of flows and the queuing delay they experience. In addition, the numerical evaluation indicated that the algorithm worked well with delay information on a few of tens of sample assignments, which made the algorithm prac-

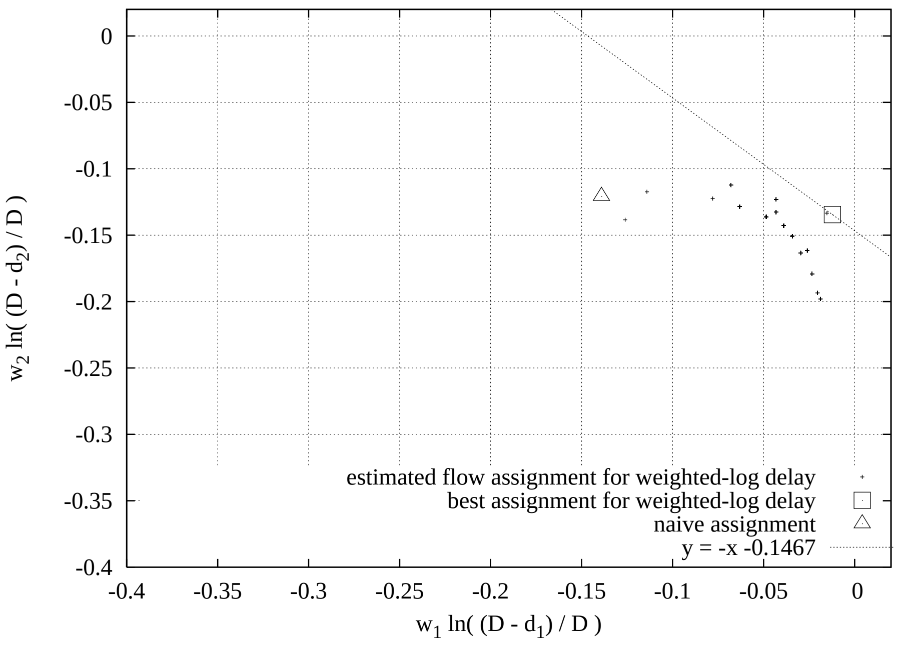


Figure 6.13: The 99.5th percentile of queuing delay for estimated flow assignment with weighted-log delay for the case of only 20 sample data and 80 % utilization ($D$ = 0.1 s)

tical in on-line use. Finally, using the above algorithm, an on-line method of distributing a number of flows with two distinct traffic characteristics over two paths for a good overall queuing delay performance was proposed.

The proposed method improved queuing delay by up to 40 % and 15 % for the min-max delay and the weighted-log delay, respectively, compared with the naive flow assignment. This work proves that taking flow-level traffic characteristics into account significantly improves delay performance in assigning flows to multiple paths, in which lower-layer information such as link bandwidth and link queuing delay is effectively combined with higher-layer information such as flow-level traffic characteristics to perform a traffic engineering for good overall delay performance.

Topics remaining as future work include developing a practical method of measuring the traffic characteristics of flows in time and devising a specific flow distribution method not only for two types of flows over two paths but also for *m* types of flows over *n* paths.

# Chapter 7

# Concluding Remarks

## 7.1 Summary of this dissertation

To improve the capacity of traffic admitted to a backbone network, distributing large amount of traffic over multiple paths is conventionally used. However, there is a nature that the constant-bit-rate flows suffer from delay variation as large as the bursty flows do when both types of flows are admitted in the single path. And this nature escalates when the paths are highly utilized, and therefore real-time applications may encounter insufficient QoS. To improve the QoS as a whole network as well as the capacity of admitted traffic, taking traffic characteristics of flows into account in distributing traffic is vital.

To confirm the effectiveness of the traffic characteristic-aware TE, the followings were studied,

I clarifying the structure of an backbone network and the functions equipped in edge routers in it,

II the availability of scalable flow measurement method for a high-speed link through developing a real-time flow monitoring tool.

III traffic characteristic-aware flow assignment reducing queuing delay for two distinct flow sets with an arbitrary traffic characteristics to be assigned to two different bandwidth paths.

I. was described in Chapter 2. II. was showed through developing the flow monitoring tool in Chapter 3, confirming the effectiveness of focusing on the flow with two analysis on TCP flows in Chapter 4 and UDP flows in Chapter 5. III. was described in Chapter 6. Through two studies in Chapters 4 and 6, this dissertation proved that introducing the traffic characteristics in the TE could improve the capacity of the admitted traffic and the QoS simultaneously. The summaries of individual chapters are follows.

Chapter 2 described Asia-Pacific Advanced Network (APAN) and reviewed the network design and operations of APAN Tokyo exchange point (XP). The network operation of APAN Tokyo XP is forced to use the policy based routing to meet different routing policies for the individual international links. The source address checking of policy based routing generally falls the packet forwarding performance due to packet processing forced in in the router main processor in which the source address checking is insufficiently optimized. To improve the processing performance with those routers policy based routing scheme with multiple routers were proposed. The evaluation showed that the proposed scheme had the sufficient packet-forwarding performance (50 Kpps for 64 IPv4 source addresses). This performance satisfied the requirement in Tokyo XP (24.4 Kpps for 19 source IPv4 addresses). Through the review of APAN Tokyo XP operation, this chapter clarified the requirements for a backbone network which improving the QoS of large volume of traffic with the traffic characteristics of flows.

Chapter 3 described the architecture for a scalable real-time flow measurement tool in order to allow network operators to flexibly define “the targeted flows” on-demand, to obtain various statistics on those flows, and to visualize them in a real-time manner. The architecture adopted multiple capture devices for processing packets in parallel, hierarchical chained flow definitions for reducing average overheads of pattern-matching, and an adaptive tree data structure for dealing with a large number of active flows. The performance evaluation with a proto-type implementation on PC-UNIX showed that the proposed architecture had the scalability and the advantage of flexible fine-grained flow measurements. For example, the maximum performance reached 80 Kpps with six capture devices even by the proto-type system on nominal PC-UNIX machines. The real-time fine-grained flow visualization enables network operators to understand what types of traffic are dominant and, at the same time, how they are behaving over a short time-scale. Monitoring bidirectional TCP flows enables ones to infer RTT behavior without sending any probe packet.

Chapter 4 described analysis of the relation between the characteristics of traffic generated by typical grid applications, and the effect of the round-trip time and bottleneck bandwidth of network on the application-level performance (i.e., completion time) of these applications. All the distributed computing applications increased their completion time and the amounts of transmitted data as the scale of problems became larger while the relationship between the amount of transmitted data and the completion time heavily depended on the applications. The analysis showed that the impact of network conditions on the performance of various applications and the impact of application traffic on network conditions also differed considerably depending on the application. In addition, the examples were showed: the information on the above mentioned network-related properties of individual applications could be used to improve the total application-level performance, when two applications ran through a shared link with a limited bandwidth.

Chapter 5 described a method for scalable performance monitoring of VoIP traffic flows distinguished by five tuples, that is, IP addresses, protocol, and port numbers. This method extracted the VoIP flows in a lightweight manner, and estimated the packet loss rate and the delay variation by exploiting the features of actual VoIP flows, that is, fixed packet length and almost fixed inter-packet gaps which were relatively common among most VoIP flows. The evaluation by simulation with trace data of actual traffic showed that monitoring only 2 % or less of the total VoIP flows could accurately estimate the delay variation of the total

VoIP flows, while 20 % or more of the total VoIP flows were required in order to estimate the packet loss rate. The reasons that few flows are sufficient for the delay variation were considered with the packet arrival process and the evaluation showed that its result was quite congruent with this consideration.

Chapter 6 described a novel traffic characteristic-aware flow assignment method to reduce queuing delay in a fundamental case, where two types of flows with distinct traffic characteristics (e.g., burstiness) were distributed to two paths. The method found a nearly optimal flow assignment in terms of minimizing the worst queuing delay among two paths. The evaluation suggests that considering the traffic characteristics significantly improves the delay performance in the flow distribution over multiple paths.

## 7.2 Issue for Future Research

In these studies, edge routers were considered to have measuring traffic characteristics of flows and balancing them over multiple paths. Underlying these studies, there is an assumption that the bandwidth-guaranteed path are appropriately placed over the core network according with the traffic dynamics. The effective cooperation of the propose flow assignment method with path placement method is also integral to the traffic characteristics-aware TE. reflecting the traffic dynamics. In addition, There are the future work for each of studies described in this dissertation.

Traffic monitoring of flows should be scalable to target the rate of 10 Gbit/s or more. To follow such rate, the number of tracked flows should be dramatically reduced with the feature of the actual traffic. E.g., large amounts of flows terminate in very short duration or large volume of flows has high correlation to being long lived.

The flow assignment method described in Chapter 6 exploited traffic with on and off state varies along the Poison process. With such an ideal traffic, the proposed flow assignment reduced the queuing delay up to 40 % againet that for the link-utility based flow assignment. The method should also be evaluated with the real traffic. Taking into account that the method deals with two types of traffic to be allocated to two path, grouped flows should have two distinct traffic characteristics. Appropriate flow segregation should be found, such as, the segregation of TCP and UDP flows, the difference of (large and small) average packet size of flows, or the difference of port speed of network interface card from which the traffic enter to an edge router.

# Bibliography

[AAA$^+$02] H. Abrahamsson, B. Ahlgren, J. Alonso, A. Andersson, and P. Kreuger, "A multi path routing algorithm for IP networks based on flow optimization", in *LNCS 2511: Proceedings of International Workshop on Quality of Future Internet Services (QoFIS)*, October 2002.

[abi] "Abilene", http://abilene.internet2.edu/.

[ACE$^+$02] Daniel Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao, "Overview and principles of Internet traffic engineering", RFC 3272, Internet Engineering Task Force, May 2002.

[AMA$^+$99] Daniel Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus, "Requirements for traffic engineering over MPLS", RFC 2702, Internet Engineering Task Force, September 1999.

[AMS82] D. Anick, D. Mitra, and M. M. Sondhi, "Stochastic theory of a data-handling system with multiple sources", *The Bell System Technical Journal*, Vol. 61, No. 8, pp. 1871–1894, 1982.

[anra] "MD1230A-20, MX123001A-20, application traffic monitor", http://www.anritsu.com/.

[anrb] “Product introduction – MD1230A familiy –”, http://www.anritsu.co.jp/Products/pdf_e/MD1230Afamily_EI1100.pdf.

[AO05] K. Aida and T. Osumi, “A case study in running a parallel branch and bound applications”, in *Proceedings of the 2005 International Symposium on Application and the Internet (SAINT2005)*, pp. 164–173, 2005.

[APH99] H. E. Bal A. Plaat and R. F. H. Hofman, “Sensitivity of parallel applications to large differences in bandwidth and latency in two-layer interconnects”, in *Proceedings of the 5th High Perfromance Computer Architecture*, pp. 244–253, 1999.

[AVL62] G. M. Adelson-Velskii and Y. M. Landis, “An algorithm for the organization of information”, *Doklady Akad. Nauk SSSR*, Vol. 146, pp. 263–266, 1962.

[BBB+91] D. H. Bailey, E. Barszcz, J. T. Barton, D. S. Browning, R. L. Carter, L. Dagurm, R. A. Fatoohi, P. O. Federickson, T. A. lasinski, R. S. Schreiber, H. D. Simon, V. Venkatakrishnan, and S. K. Weeratunga, “The NAS parallel benchmarks”, *International Journal of Supercomputer Applications*, Vol. 5, No. 3, pp. 66–73, 1991.

[BD] Carter Bullard and Chas DiFatta, “Argus”, Technical report, Software Engineering Institute, Carnegie Mellon University, http://www.qosient.com/argus/.

[BIY01] Tomomitsu BABA, Hidetaka IZUMIYAMA, and Suguru YAMAGUCHI, “AI$^3$ satellite Internet infrastructure and the deployment in Asia”, *IEICE Transactions.Communication*, No. 8, pp. 2058–2065, August 2001.

[BRCK00] T. Bates, Y. Rekhter, R. Chandra, and D. Katz, “Multiprotocol extensions for BGP-4”, RFC 2858, Internet Engineering Task Force, June 2000.

[Bro97] Nevil Brownlee, "Traffic flow measurement: Experiences with NeTraMet", RFC 2123, Internet Engineering Task Force, March 1997.

[Buy99] R. Buyya, *High Performance Cluster Computing: Programming and Applications*, Vol. 2, Prentice Hall PTR, 1999.

[cai] "CAIDA", http://www.caida.org/.

[Che99] J. Chen, *New Approaches to Routing for Large-Scale Data Networks*, PhD thesis, June 1999, Adviser-Richard A. Thompson.

[Cla82] D. Clark, "Window and acknowledgement strategy in TCP", RFC 813, Internet Engineering Task Force, July 1982.

[DCS01] Luca Deri, R. Carbone, and S Suin, "Monitoring networks using ntop", in *Proceedings of IM 2001*, pp. 199–212, May 2001.

[Def01] Thomas A. Defanti, "StarTAP2: Science, technology and research transit access point annual report", Technical report, 2001, http://www.startap.net/startap/images/PDF/STARTAP2.AnnualReport2001.FINAL.3.5.01a.pdf.

[Der03] Luca Deri, "nProbe: an open source NetFlow probe for gigabit networks", in *Proceedings of TERENA Networking Conference 2003*, May 2003.

[Der06] Luca Deri, "Open source VoIP traffic monitoring", in *Proceedings of SANE 2006*, May 2006.

[DLT01] N. Duffield, C. Lund, and M. Thorup, "Charging from sampled network usage", in *Proceedings of ACM SIGCOMM Internet Measurement Workshop*, pp. 245–256, November 2001.

[DLT02] N. Duffield, C. Lund, and M. Thorup, "Properties and prediction of flow statistics from sampled packet streams", in *Proceedings of ACM SIGCOMM Internet Measurement Workshop*, pp. 159–171, November 2002.

[DTVV90] Maurizio Decina, T. Toniatti, P. Vaccari, and L. Verri, "Bandwidth assignment and virtual call blocking in ATM networks", in *Proceedings of IEEE INFOCOM*, pp. 881–888, 1990.

[EFH$^+$98] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, and C. Liu, "Protocol independent multicast-sparse mode (PIM-SM): protocol specification", RFC 2362, Internet Engineering Task Force, June 1998.

[EJLW01] A. Elwalid, C. Jin, S. Low, and I. Widjaja, "MATE: MPLS adaptive traffic engineering", in *Proceedings of IEEE INFOCOM*, pp. 1300–1309, April 2001.

[ets02] "End-to-end quality of service in TIPON systems; Part7: Design guide for elements of a TIPHON connection from an end-to-end speech transmission performance point of view", ETSI TS 101 329-7, European Telecommunications Standards Institute, 2002.

[FK98] I. Foster and C. Kesselman, *The GRID Blueprint for a New Computing Infrastructure,* Morgan Kaufmann Publishers, 1998.

[FKT01] I. Foster, C. Kesselman, and S. Tuecke, "The anatomy of the Grid: Enabling scalable virtual organizations", *International Journal of Supercomputer Applications,* Vol. 15, No. 3, pp. 200–222, 2001.

[FM03] B. Fenner and D. Meyer, "Multicast source discovery protocol (MSDP)", RFC 3618, Internet Engineering Task Force, October 2003.

[FT00] B. Fortz and M. Thorup, "Internet traffic engineering by optimizing OSPF weights", in *Proceedings of IEEE INFOCOM*, pp. 519–528, 2000.

[g1004] "The E-model, a computational model for use in transmission planning", Telecommunication Standardization Sector of International Telecommunication Union Recommendation G. 107, International Telecommunication Union, 2004.

[g7198] "Pulse code modulation (PCM) of voice frequencies", Telecommunication Standardization Sector of International Telecommunication Union Recommendation G. 711, International Telecommunication Union, 1998.

[g7296] "Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear-prediction", Telecommunication Standardization Sector of International Telecommunication Union Recommendation G.729, International Telecommunication Union, 1996.

[GCL04a] K. Gopalan, T. Chiueh, and Y.-J. Lin, "Load balancing routing with bandwidth-delay guarantees", *IEEE Communications*, Vol. 42, No. 6, pp. 108–113, 2004.

[GCL04b] K. Gopalan, T. Chiueh, and Y.-J. Lin, "Probabilistic delay guarantees using delay distribution measurement", in *Proceedings of the 12th annual ACM international conference on Multimedea*, pp. 900–907, October 2004.

[GKL$^+$04] T. Guven, C. Kommareddy, R. J. La, M. A. Shayman, and B. Bhattacharjee, "Measurement based optimal multi-path routing", in *Proceedings of IEEE INFOCOM*, pp. 187–196, March 2004.

[GT] M. Grant and S. Tenissen, "Voice quality monitoring for VoIP networks", White paper, http://www.calyptech.com/pdf/CAL-000006-WP-01.pdf.

[icp01] "The ACM international collegiate programming contest Japan domestics", 2001, Probrem C Jigsaw Puzzle for Computers (http://www.fun.ac.jp/icpc/domestic_problems.html).

[inm] "InMon Corporation", http://www.inmon.com/.

[ipe] "Iperf - the TCP/UDP bandwidth measurement tool", http://dast.nlanr.net/Projects/Iperf/.

[Kaw04] Ryoichi Kawahara, "An adaptive load balancing method for multiple paths using flow statistics and its performance analysis", *IEICE Transactions on Communications*, No. 7, pp. 1993–2003, 2004.

[kis] "KISDI", http://www.kisdi.re.kr/eng_kisdi/.

[KKK$^{+}$02] Yoshinori Kitatsuji, Katsushi Kobayashi, Yasuichi Kitamura, Akira Kato, and Kazunori Konishi, "Development of APAN Tokyo XP and evaluation of source-based routing", *IEICE Transactions on Communications*, Vol. J85-B, No. 8, pp. 1164–1171, August 2002.

[KKL00] K. Kar, M. Kodialam, and T.V. Lakshman, "Minimum interference routing of bandwidth guaranteed tunnels with MPLS traffic engineering applications", *IEEE Journal of Selected Areas in Communications*, Vol. 18, No. 12, pp. 2566–2579, 2000.

[KKT+06] Yoshinori Kitatsuji, Satoshi Katsuno, Masato Tsuru, Tetsuya Takine, and Yuji Oie, "On flow distribution over multiple paths based on traffic characteristics", in *LNCS 3961, ICOIN 2006*, pp. 483–492. Springer-Verlag Berlin Heidelberg, January 2006.

[KKT+07] Yoshinori Kitatsuji, Satoshi Katsuno, Masato Tsuru, Tetsuya Takine, and Yuji Oie, "Traffic characteristics-based flow assignment method for reducing queuing delay", April 2007, to appear in Proceedings of International Conference on Networkings 2007 (ICN'07).

[KKY+07] Yoshinori Kitatsuji, Satoshi Katsuno, Katsuyuki Yamazaki, Masato Tsuru, and Yuji Oie, "A distributed flow monitoring tool using network processor", in *Proceeding of 2007 International Symposium on Applications and the Internet Workshops (SAINTW'07)*, pp. 87–90, January 2007.

[KMK+01] Ken Keys, David Moore, Ryan Koga, Edouard Lagache, Michael Tesch, and K claffy, "The architecture of CoralReef: an Internet traffic monitoring software suite", in *PAM2001 — A workshop on Passive and Active Measurements*. CAIDA, RIPE NCC, April 2001, http://www.caida.org/tools/measurement/coralreef/.

[kor] "KOREN", http://www.koren21.net/english/.

[KSOT] T. Kitamura, T. Shizuno, T. Okabe, and H. Tani, "Traffic identification for dependable VoIP", *NEC Technical Journal*, Vol. 1, No. 3.

[KWW+98] S. Knight, D. Weaver, D. Whipple, R. Hinden, D. Mitzel, P. Hunt, P. Higginson, and M. Shand, "Virtual router redundancy protocol", RFC 2338, Internet Engineering Task Force, April 1998.

[KYK$^+$05] Yoshinori Kitatsuji, Katsuyuki Yamazaki, Hiroshi Koide, Masato Tsuru, and Yuji Oie, "Influence of network characteristics on application performance in a Grid environment", *Telecommunication Systems*, Vol. 30, No. 1/2/3, pp. 99–121, November 2005, Monufactured in The Netherlands.

[KYTO04] Yoshinori Kitatsuji, Katsuyuki Yamazaki, Masato Tsuru, and Yuji Oie, "Real-time IP flow measurement tool with scalable architecture", *IEICE Transactions on Information and Systems*, Vol. E87-D, No. 12, pp. 2665–2676, December 2004.

[LW00] Francis Lee and C.Y. Wong, "APAN documentation", Technical report, Asia-Pacific Advanced Network, 2000, https://www.singaren.net.sg/apps/apan_doc/contents.html.

[McR] Daniel McRobb, "cFlowd: Traffic flow analysis tool", http://www.caida.org/tools/measurement/cflowd/.

[Mil85] D. Mills, "Network time protocol (NTP)", RFC 958, Internet Engineering Task Force, September 1985.

[MJ93] S. McCanne and V. Jacobson, "The BSD packet filter: A new architecture for user-level packet capture", January 1993.

[MKK$^+$] David Moore, Ken Keys, Ryan Koga, Edouard Lagache, and k claffy, "The CoralReef software suite as a tool for system and network administrators".

[mrt] "Multi router traffic grapher", http://www.mrtg.org/.

[net02] "NetFlow service and applications", White paper, Cisco Systems, Inc., 2002, http://www.cisco.com/warp/public/cc/
pd/iosw/ioft/neflct/tech/napps_wp.htm.

[NYS+98] L. H. Ngoh, F. S. C. Yeoh, M. Singh, C. K. Tham, A. L. Ananda, T. H. Cheng, and B. S. Lee, "SingAREN: The singapore advanced research and education network", *IEEE Communications Magazine*, November 1998.

[Oet] Tobias Oetiker, "Round robin database tool (RRDTool)", http://people.ee.ethz.ch/ oetiker/webtools/rrdtool/.

[p5604] "Single-ended method for objective speech quality assessment in narrowband telephony applications", Telecommunication Standardization Sector of International Telecommunication Union Recommendation P. 563, International Telecommunication Union, 2004.

[pac] "Empirix packet sphere", http://www.empirix.com/.

[PDC+06] M. Pickavet, P. Demeester, D. Colle, D. Staessens, B. Puype, L. Depre, and I. Lievens, "Recovery in multilayer optical networks", *IEEE Journal of Lightwave Technology*, Vol. 24, No. 1, pp. 122–134, 2006.

[Plo00] Dave Plonka, "Flowscan: Network traffic flow visualization and reporting tool", in *Proceedings of the Fourteenth Systems Administration Conference (LISA-00)*, pp. 305–318, December 2000, http://www.caida.org/tools/utilities/flowscan/.

[PPM01] P. Phaal, S. Panchen, and N. McKee, "InMon corporation's sFlow: A method for monitoring traffic in switched and routed networks", RFC 3176, Internet Engineering Task Force, September 2001.

[QZCZ04] J. Quittek, Tanja Zseby, B. Claise, and S. Zander, "Requirements for IP flow information export (IPFIX)", RFC 3917, Internet Engineering Task Force, October 2004.

[Ram99] V. Ramaswami, “Matrix analytic methods for stochastic fluid flows”, in *Proceedings of ITC 16*, pp. 1019–1030, 1999.

[Rao01] N. S. V. Rao, “NetLets: End-to-end QoS mechanisms for distributed computing in wide-area networks using two-paths”, in *Proceedings of the first International Conference on Internet Computing*, pp. 475–478, 2001.

[RL95] Y. Rekhter and T. Li, “A border gateway protocol 4 (BGP-4)”, RFC 1771, Internet Engineering Task Force, March 1995.

[SB02] P. Siripongwutikorn and S. Banerjee, “Per-flow delay performance in traffic aggregates”, in *Proceedings of IEEE Globecom*, Vol. 21, pp. 2641–2645, 2002.

[SCFJ96] Henning Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, “RTP: a transport protocol for real-time applications”, RFC 1889, Internet Engineering Task Force, January 1996.

[SdV01] X. Su and G. de Veciana, “Dynamic multi-path routing: asymptotic approximation and simulations”, *ACM SIGMETRICS Performance Evaluation Review*, Vol. 29, No. 1, pp. 25–36, 2001.

[SE03] Vinod Sharma and F. (Editors), “Framework for multi-protocol label switching (MPLS)-based recovery”, RFC 3469, Internet Engineering Task Force, February 2003.

[SGD03] A. Sridharan, R. Guerin, and C. Diot, “Achieving near-optimal traffic engineering solutions for current OSPF/IS-IS networks”, in *Proceedings of IEEE INFOCOM*, pp. 1167–1177, 2003.

[Son01] Jinseok Song, *Spare capacity reduction using vertical sharing in multilayer mesh restorable network*, PhD thesis, 2001, Adviser-Richard A. Thompson.

[TK02] T. Tobita and H. Kasahara, "A standard task graph set for fair evaluation of multiprocessor scheduling algorithms", *Journal of Scheduling*, Vol. 5, No. 5, pp. 379–394, 2002.

[Top] D. M. Topkis, "A k-shortest-path algorithm for adaptive routing in communications networks".

[tra02] "TransPAC annual report 2001-2002", Technical report, Indiana University, TransPAC, 2002, http://transpac.org/old-site/documents/2002_transpac_ar.pdf.

[TV00] V. Trecordi and G. Verticale, "Per-flow delay performance in a FIFO scheduler fed by policed UDP sources", *Computer Communications*, Vol. 23, No. 4, pp. 309–316, 2000.

[Vil99] C. Villamizar, "OSPF optimized multipath (OSPF-OMP)", Expired internet draft, 1999, Available at http://www.ietf.org/proceedings/99mar/I-D/draft-ietf-ospf-omp-02.txt.

[vip] "Versatile interface processor 2 for Cisco 7500 series routers", Technical report, Cisco Systems, Inc., http://www.cisco.com/warp/public/cc/pd /ifaa/ifpz/vrifpz/prodlit/vip2_ds.htm.

[Wal00] S. Waldbusser, "Remote network monitoring management information base", RFC 2819, Internet Engineering Task Force, May 2000.

[whi06] "Information and communications in Japan", White paper, Ministry of Internal Affairs and Communications, 2006, http://www.johotsusintokei.soumu.go.jp/whitepaper/eng/WP2006/ 2006-index.html.

[wid] “WIDE project”, http://www.wide.ad.jp/.

[WK] Thomas Williams and Colin Kelley, “GNUPLOT homepage”, http://gnuplot.info/.

[XG02] Y. Xu and R. Guerin, “Individual QoS versus aggregate QoS: A loss performance study”, in *Proceedings of IEEE INFOCOM*, pp. 1170–1179, June 2002.

[ZL03] M. Zangrilli and B. B. Lowekamp, “Comparing passive network monitoring of GRID application traffic with active probes”, in *Proceedings of Fourth International Workshop on Grid Computing*, pp. 84–91, November 2003.

[ZWJ03] A. Zeitoun, Z. Wang, and S. Jamin, “RTTometer: Measuring path minimum RTT with confidence”, in *Proceedings of IEEE Workshop on IP Operations and Management (IPOM 2003)*, October 2003.