



Title	Studies on Adaptive Routing for a Wide-Area Overlay Network System
Author(s)	柏崎, 礼生
Citation	北海道大学. 博士(情報科学) 乙第6907号
Issue Date	2014-03-25
DOI	10.14943/doctoral.r6907
Doc URL	<a href="http://hdl.handle.net/2115/55690">http://hdl.handle.net/2115/55690</a>
Type	theses (doctoral)
File Information	Hiroki_Kashiwazaki.pdf



[Instructions for use](#)

DOCTORAL THESIS

---

# Studies on Adaptive Routing for a Wide-Area Overlay Network System

---

Hiroki KASHIWAZAKI

*A thesis submitted in fulfilment of the requirements  
for the degree of Doctor of Information Science*

January 2014

# *Abstract*

## **Studies on Adaptive Routing for a Wide-Area Overlay Network System**

by Hiroki KASHIWAZAKI

This paper proposes and evaluates two types of traffic engineering algorithms. The first, REI, and its derivative, NREI, are decentralized stochastic methods that have autonomous adaptability to network traffic conditions. When a routing node has some different paths to a given destination, these paths can be evaluated in terms of their latency (delay time) information, which will be given from a feedback packet sent back from a destination node to a source node after an outbound packet reaches the destination node. This algorithm is based on a one-way delay algorithm. NREI was developed from the delay time based adaptive routing algorithm, REI, and is enforced for packet losses caused by network congestion. Based on a path evaluation using latency and penalty scores from lost packets, every node works as a distributed autonomous agent for adaptive routing. Through network simulations comparing these algorithms to conventional and enhanced OSPFs and evaluations on an IP network of a research test bed, it is shown that a multi-agent based routing algorithm has better adaptability and scalability to congested path avoidance and network load balancing. In addition, the other approach proposed is a centralized deterministic method that uses a discrete event simulator. This method searches for a suboptimal combination of paths from the enormous number of combinations of paths determined from pairs of all source nodes and all destination nodes. Cloud computing resources are used to evaluate these huge combinations rapidly, by which we can utilize powerful computing resources. This method was tested on an ns-2 simulator with eight- and eleven-node networks. The results show that the method has better adaptability and high-speed performance for cross-traffic avoidance. Finally, implementation using OpenFlow is considered and herein a hybrid model consisting of the two proposed algorithms is discussed.

# Contents

<b>Abstract</b>	<b>i</b>
<b>List of Figures</b>	<b>iv</b>
<b>List of Tables</b>	<b>vi</b>
<b>1 Introduction</b>	<b>1</b>
1.1 The Internet-based communication traffic	1
1.1.1 Growth of Internet video	3
1.1.2 Large Traffic from Mobile, Sensing Devices	5
1.1.3 Big Data	6
1.2 Multihome Network	9
1.2.1 Overlay Networking	11
1.3 Traffic Engineering	12
1.3.1 Active Networking	14
1.3.2 Software-defined Networking	14
1.3.3 OpenFlow	17
1.4 Objective	20
<b>2 Decentralized Stochastic Approach</b>	<b>23</b>
2.1 Introduction	23
2.2 Approach by Distributed Routing Agent	24
2.2.1 Network latency	24
2.2.2 Routing tables	24
2.2.3 Next hop selection	25
2.3 Experimental Evaluation by Simulations	27
2.3.1 Comparative approaches	27
2.3.2 Experimental settings	27
2.3.3 Uniform traffic condition	28
2.3.4 Concentrated traffic condition	30
2.3.5 Network disturbance condition	31
2.3.6 Effectiveness of parameter $\lambda$	32
2.3.7 Evaluation on a scale-free network	35
2.3.8 Load balancing for network resources	36
2.4 Experimental Evaluation in IP network	42
2.4.1 A routing table and a probe packet	42
2.4.2 Algorithm for the next node selection	44

2.4.3	Experimental settings	46
2.4.4	Experiment 1. Raising the total amount of traffic demands	49
2.4.5	Experiment 2. Raising the delay on one side	51
<b>3</b>	<b>Centralized deterministic approach</b>	<b>54</b>
3.1	Objective	54
3.2	Basic Framework	55
3.3	Evaluation Calculation for a Combination of Paths	58
3.3.1	Collection of Network Information	58
3.3.2	Control Service	59
3.3.3	Scoring of Paths	61
3.3.4	Evaluation Calculation by Simulation	63
3.4	Examinations	64
3.4.1	Accuracy Evaluations of the Simulator	64
3.4.2	Computation Time Evaluations of Simulator	65
3.4.3	Evaluation Examination	67
<b>4</b>	<b>Implementation and Discussion</b>	<b>71</b>
4.1	Discussion Regarding the Performance	71
4.1.1	Decentralized Stochastic Approach	71
4.1.2	Centralized Deterministic Approach	72
4.2	Implementation on OpenFlow	74
4.2.1	History of OpenFlow	74
4.2.2	Specifications of OpenFlow	74
4.3	Implementation of the decentralized stochastic approach on OpenFlow	78
4.3.1	xFlow and sFlow	78
4.3.2	Overview of the implementation	79
4.3.3	Procedure in the implementation	80
4.4	Implementation of the centralized deterministic approach on OpenFlow	81
4.4.1	Overview and procedure of the implementation	82
4.5	Hybrid Approach	83
<b>5</b>	<b>Conclusion</b>	<b>87</b>
5.1	Summary	87
5.2	Future Works	89

# List of Figures

1.1	An Actual Status of the Internet Traffic in Japan. . . . .	1
1.2	Global IP Traffic Growth, 2012-2017 (Source: Cisco VNI, 2013). . . . .	2
1.3	Consumer Internet Traffic, 2012-2017 (Source: Cisco VNI, 2013). . . . .	3
1.4	Contents Delivery Network Traffic vs Consumer IP Traffic, 2012-2017 (Source: Cisco VNI, 2013). . . . .	4
1.5	Changes in the maintenance conditions for the broadband connection platform in Japan (source: 平成 25 年版情報通信白書 (総務省) licensed under CC-BY 2.1 JP). . . . .	8
1.6	Fixed broadband spread ratio, FTT ratio, and per capita Internet spread ratio. . . . .	10
1.7	Diagram of multihoming. . . . .	10
2.1	Congestion avoidance. . . . .	23
2.2	A REI routing table with scores. . . . .	25
2.3	Probabilistic selection of the next node. . . . .	26
2.4	Network models for the simulation experiments. . . . .	27
2.5	Average latency on the uniform traffic condition. . . . .	28
2.6	Route distribution ratio of REIsx. . . . .	29
2.7	Background traffic and gradient of latency. . . . .	30
2.8	Concentrated traffic and gradient of latency. . . . .	31
2.9	Average latency on the traffic disturbance. . . . .	32
2.10	Route distribution of REIsx on the traffic disturbance. . . . .	33
2.11	Average latency in SFN. . . . .	34
2.12	Gradient of average latency in SFN. . . . .	35
2.13	Distribution of queued packets with OSPF in Random Network. . . . .	37
2.14	Distribution of queued packets in nodes. (OSPF) . . . . .	37
2.15	Distribution of queued packets with OSPF+ in Random Network. . . . .	38
2.16	Distribution of queued packets in nodes. (OSPF+) . . . . .	38
2.17	Distribution of queued packets with OSPF++ in Random Network. . . . .	39
2.18	Distribution of queued packets in nodes. (OSPF++) . . . . .	39
2.19	Distribution of queued packets with REIsx ( $\lambda = 4.0$ ) in Random Network. . . . .	40
2.20	Distribution of queued packets with OSPF++ in SFN. . . . .	41
2.21	Distribution of queued packets with REIsx ( $\lambda = 4$ ) in SFN. . . . .	41
2.22	Routing table with scores. . . . .	43
2.23	Protocol for probe packets and updating routing table. . . . .	44
2.24	Probabilistic selection of the next-node. . . . .	45
2.25	Network construction for evaluation experiments. . . . .	48
2.26	Changes of delay and packet loss ratio for increment of network traffic . . . . .	49

2.27	Changes of packet distribution for path delay increment (3 Mbps)	50
2.28	Changes of packet distribution for path delay increment (4 Mbps)	51
2.29	Changes of packet distribution for path delay increment (5 Mbps)	52
3.1	Overlay Network Topology.	56
3.2	Rerouting for Overlay Network.	57
3.3	Measuring of Network Information.	60
3.4	Path Scoring Based on Total Latency.	62
3.5	Making Combinations of paths and Evaluation.	63
3.6	Networks for the Simulator Evaluation.	65
3.7	Convergence to Optimal Solutions in the Barabási-Albert Model Topology.	66
3.8	Convergence to Optimal Solutions in the Waxman Model Topologies.	67
3.9	Network for Evaluation Experiment	68
3.10	Change of Packet Loss.	69
3.11	Change of Packet Loss and Evaluation for Combination of Paths.	69
4.1	Main components of an OpenFlow switch.	75
4.2	Overview of the implementation of the decentralized stochastic approach	80
4.3	Procedure of the implementation of the decentralized stochastic approach	82
4.4	Procedure of the implementation of the centralized deterministic approach	83
4.5	Overview of the hybrid approach	85

# List of Tables

2.1	$\lambda$ effect on the latency distribution. . . . .	33
3.1	Relative Precision of the Network Simulator to ns-2. . . . .	65

# Chapter 1

## Introduction

### 1.1 The Internet-based communication traffic

The widespread nature of Internet-based communication and increasing richness of Internet content is bringing about constant increases in upload and download network traffic. In Japan alone, according to a September, 2012 report from the Ministry of Internal Affairs and Communications of Japan, the total amount of download traffic from broadband Internet subscribers was 1.7 Tbps as of May, 2012, and the amount of upload traffic was 658 Gbps (Fig. 1.1). In 2009, a revised copyright act came into force, and this law made it illegal to download illegally uploaded content when its illegality

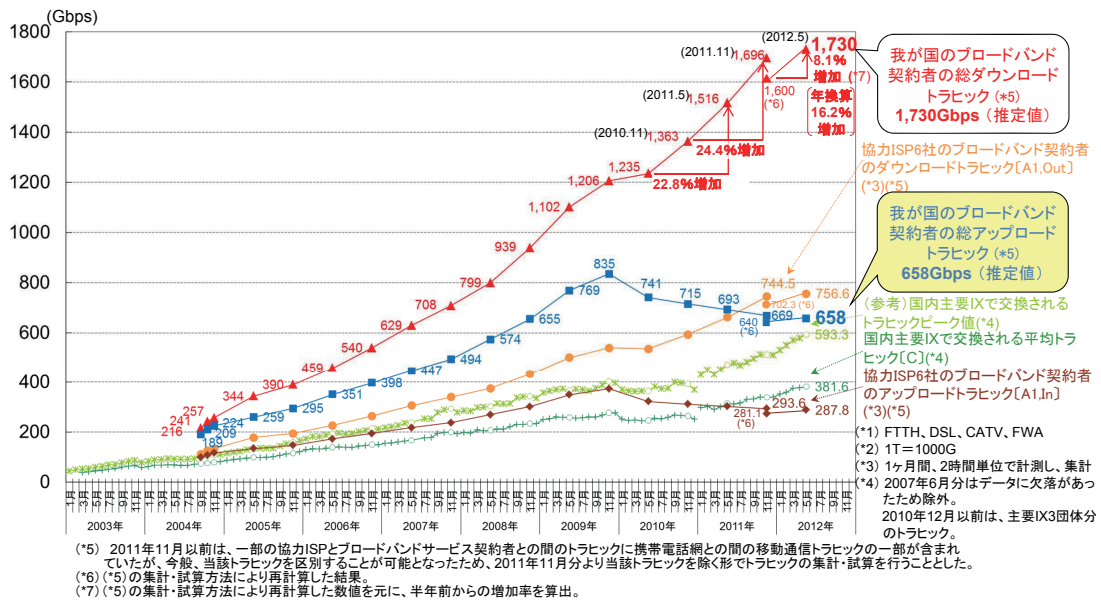


FIGURE 1.1: An Actual Status of the Internet Traffic in Japan.

IP Traffic, 2012–2017							
	2012	2013	2014	2015	2016	2017	CAGR 2012–2017
<b>By Type (PB per Month)</b>							
Fixed Internet	31,339	39,295	47,987	57,609	68,878	81,818	21%
Managed IP	11,346	14,679	18,107	21,523	24,740	27,668	20%
Mobile data	885	1,578	2,798	4,704	7,437	11,157	66%
<b>By Segment (PB per Month)</b>							
Consumer	35,047	45,023	56,070	68,418	82,683	98,919	23%
Business	8,522	10,530	12,822	15,417	18,372	21,724	21%
<b>By Geography (PB per Month)</b>							
Asia Pacific	13,906	18,121	22,953	28,667	35,417	43,445	26%
North America	14,439	18,788	23,520	28,667	34,457	40,672	23%
Western Europe	7,722	9,072	10,568	12,241	14,323	16,802	17%
Central and Eastern Europe	3,405	4,202	5,167	6,274	7,517	8,844	21%
Latin America	3,397	4,321	5,201	5,975	6,682	7,415	17%
Middle East and Africa	701	1,049	1,483	2,013	2,659	3,465	38%
<b>Total (PB per Month)</b>							
Total IP traffic	43,570	55,553	68,892	83,835	101,055	120,643	23%

FIGURE 1.2: Global IP Traffic Growth, 2012-2017 (Source: Cisco VNI, 2013).

is known. Largely due to this act, it was reported that the amount of upload traffic decreased for the first time ever since observations began. Through a revision in the law, the number of P2P users has dramatically decreased, as has the amount of upload traffic. However, the amount of download traffic has been increasing since the revision. The amount of download traffic in May, 2012 was 1.1-times larger (8.1 % increase) than the amount in November 2011, and was 1.2-times larger (16.2 % increase) than in May 2011<sup>1</sup> [1].

According to Cisco Visual Networking Index (VNI): Forecast and Methodology, 2012-2017, the amount of global Internet traffic per month was 43.6 exabytes in 2012, and is expected reach 120.6 exabytes in 2017 (Fig. 1.2)<sup>2</sup>. This supposes that consumer IP traffic per month will reach 98.9 exabytes and business traffic will exceed 21.7 exabytes. In this supposition, consumer traffic is from households, university populations, and Internet cafes generating a fixed amount of IP traffic, whereas business traffic is from businesses and governments generating a fixed amount of IP WAN or Internet traffic. The amount of video content on the Internet exceeded half of the global consumer traffic by the end of 2011. The total amount of all video forms (TV, video on demand, Internet, and P2P) is estimated to become approximately 69 % of global consumer traffic by 2017. This asymmetric growth in the amount of download and upload traffic can be ascribed to the rising amount of rich content such as video communication, high-density (HD) movie streaming, and big data.

<sup>1</sup>[http://www.soumu.go.jp/menu\\_news/s-news/01kiban04\\_02000042.html](http://www.soumu.go.jp/menu_news/s-news/01kiban04_02000042.html)

<sup>2</sup>[http://www.cisco.com/en/US/netsol/ns827/networking\\_solutions\\_sub\\_solution.html](http://www.cisco.com/en/US/netsol/ns827/networking_solutions_sub_solution.html)

Consumer Internet Traffic, 2012–2017							
	2012	2013	2014	2015	2016	2017	CAGR 2012–2017
<b>By Network (PB per Month)</b>							
Fixed	25,529	32,097	39,206	47,035	56,243	66,842	21%
Mobile	684	1,239	2,223	3,774	6,026	9,131	68%
<b>By Subsegment (PB per Month)</b>							
Internet video	14,818	19,855	25,800	32,962	41,916	52,752	29%
Web, email, and data	5,173	6,336	7,781	9,542	11,828	14,494	23%
File sharing	6,201	7,119	7,816	8,266	8,478	8,667	7%
Online gaming	22	26	32	39	48	59	22%
<b>By Geography (PB per Month)</b>							
Asia Pacific	9,033	11,754	14,887	18,707	23,458	29,440	27%
North America	6,834	8,924	11,312	14,188	17,740	21,764	26%
Western Europe	5,086	5,880	6,804	7,810	9,197	10,953	17%
Central and Eastern Europe	2,194	2,757	3,433	4,182	5,015	5,897	22%
Latin America	2,656	3,382	4,049	4,588	5,045	5,487	16%
Middle East and Africa	410	640	944	1,334	1,816	2,432	43%
<b>Total (PB per Month)</b>							
Consumer Internet traffic	26,213	33,337	41,429	50,809	62,269	75,973	24%

FIGURE 1.3: Consumer Internet Traffic, 2012-2017 (Source: Cisco VNI, 2013).

### 1.1.1 Growth of Internet video

Video streaming on the Internet is said to have started in the 1990s. Severe Tire Damage, a rock and roll band from Palo Alto, California, broadcast a live performance at Xerox PARC on the Internet on June 24, 1993. This was the first live performance on the Internet and it used Mbone (multicast backbone), an experimental backbone for IP multicast traffic across the Internet developed in early 1992, which was an outgrowth of the first two IETF audiocast experiments. In April, 1995, RealNetworks introduced its RealAudio Player, which is known as one of the first media players capable of streaming media content over the Internet. In March, 1996, Microsoft announced ActiveMovie, a streaming media technology distributed through the Internet. On June 8, 1999, Apple released QuickTime 4.0, which was the first release of QuickTime with streaming support. RealNetworks, Microsoft, and Apple each invented its own codec.

In March, 2002, Macromedia Flash Player 6 supported on-demand, live audio and video streaming using RTMP (Real Time Messaging Protocol)<sup>3</sup>, and began supporting Flash Video. The compression formats for Flash Video were open. Screen Video and Sorenson Spark are subsets of the H.263 compression format. Flash Player 8 supported the On2 VP6 codec, which was a proprietary codec initially, and later became open source. After 2006, Flash video became the de facto standard of Web video platforms owing to its

<sup>3</sup>[http://www.adobe.com/content/dam/Adobe/en/devnet/rtmp/pdf/rtmp\\_specification\\_1.0.pdf](http://www.adobe.com/content/dam/Adobe/en/devnet/rtmp/pdf/rtmp_specification_1.0.pdf)

CDN Traffic, 2012–2017							
	2012	2013	2014	2015	2016	2017	CAGR 2012–2017
<b>By Geography (PB per Month)</b>							
North America	4,630	6,484	9,127	12,349	16,581	21,766	36%
Asia Pacific	2,468	3,347	4,617	6,444	8,876	12,065	37%
Western Europe	2,792	3,517	4,542	5,723	7,298	9,323	27%
Central and Eastern Europe	437	586	809	1,163	1,611	2,150	38%
Latin America	465	597	747	967	1,204	1,470	26%
Middle East and Africa	71	103	142	197	265	351	38%
<b>Total (PB per Month)</b>							
CDN Internet traffic	10,863	14,633	19,984	26,842	35,834	47,124	34%

Consumer IP Traffic, 2012–2017							
	2012	2013	2014	2015	2016	2017	CAGR 2012–2017
<b>By Type (PB per Month)</b>							
Internet	25,529	32,097	39,206	47,035	56,243	66,842	21%
Managed IP	8,835	11,686	14,640	17,609	20,414	22,946	21%
Mobile data	684	1,239	2,223	3,774	6,026	9,131	68%
<b>By Geography (PB per Month)</b>							
North America	12,392	16,237	20,378	24,876	29,891	35,223	23%
Asia Pacific	10,957	14,363	18,236	22,887	28,465	35,156	26%
Western Europe	6,153	7,204	8,394	9,681	11,342	13,344	17%
Central and Eastern Europe	2,360	3,013	3,809	4,710	5,714	6,787	24%
Latin America	2,763	3,547	4,285	4,902	5,423	5,939	17%
Middle East and Africa	423	658	968	1,362	1,848	2,469	42%
<b>Total (PB per Month)</b>							
Consumer IP traffic	35,047	45,023	56,070	68,418	82,683	98,919	23%

FIGURE 1.4: Contents Delivery Network Traffic vs Consumer IP Traffic, 2012-2017  
(Source: Cisco VNI, 2013).

adoption by YouTube. On April 18, 2005, Adobe Systems announced an agreement to acquire Macromedia. Adobe Flash Player 9, Update 3, which was released in December 2007, supported H.264/MPEG-4 AVC. H.264/MPEG-4 AVC was to be widely used by Internet streaming sources, Web applications, and various HDTV broadcasts over terrestrial, cable, and satellite networks.

Fig. 1.3 shows measured and forecast data on consumer Internet traffic from 2012 to 2017. This forecast supposes that Internet video streaming and downloads are beginning to take a larger share of bandwidth, and will grow to more than 52.75 exabytes per month, which is equal to 69 % of all consumer Internet traffic expected in 2017. In the table, Internet videos include short-form Internet videos (for example, YouTube), long-form Internet videos (for example, Hulu), live Internet videos, Internet-video to TV (for example, Netflix using a Roku or other device), online video purchases and rentals, webcam viewing, and web-based video monitoring (excluding P2P video file downloads).

As with YouTube videos, the traffic amount of Internet content characterized as uploaded once, downloaded many times can be reduced using content delivery (or distribution) network (CDN) services provided by Akamai Technologies, CDNetworks, etc. A content delivery network is a large distributed system consisting of servers deployed in multiple wide-area separated data centers connected with each other through the Internet. With the appearance of popular video-streaming services, CDNs have prevailed as a dominant method for delivering such rich content. Fig. 1.4 shows that 51 % of all Internet traffic will use a CDN in 2017, which is up from 34 % in 2012 globally. The figure also shows that 65 % of all Internet video traffic will use content delivery networks in 2017, which is up from 53 % in 2012 globally.

On the other hand, content that is uploaded once, downloaded once such as high-density, unicast video conferencing for multiple-user communication using Polycom and other services is included as consumer-IP traffic. It is difficult for such content to be reduced in terms of traffic amount using a CDN owing to a difficulty in effective caching. Fig. 1.4 also shows that global consumer IP traffic is expected to reach 98.9 exabytes per month in 2017.

### 1.1.2 Large Traffic from Mobile, Sensing Devices

One of major reasons why the network traffic increased is widely spreading of mobile devices such as smart phones. Fiber optic cables that network carriers laid is not only used by fixed connection service to the Internet but also by applications of the smart phones that require connectivity to the Internet. Although the smart phones can connect to the Internet not only by 3G or 4G communication but also by wireless ethernet connection, users of the phones tend to use by wireless ethernet because of its cost and speed. Inevitably, network carriers build many of base stations of wireless ethernet especially in urban districts. Consequently, wireless network in urban districts become spreading[2].

Now we can collect physical and environmental information in the urban districts by using the wireless network and spatially distributed autonomous sensors. In general, wireless network consisted of such sensors is called a wireless sensor network. The physical and environmental information mean such as temperature, volume of sound and atmospheric pressure. The sensors periodically record and a node of the sensor send their data to servers. A node is a small computer such as Arduino<sup>4</sup> or Raspberry Pi<sup>5</sup>. The node that consists of the computer and sensor is called a sensor node. Typically, a sensor node consists of many parts such as a radio transceiver with an internal antenna or

---

<sup>4</sup><http://www.arduino.cc>

<sup>5</sup><http://www.raspberrypi.org>

connection to an external antenna, a microcontroller, an electronic circuit for interfacing with the sensors, and an energy source, usually a battery or an embedded form of energy harvesting.

When a sensors node is far from a base station of wireless network, the node pass and receive the data cooperatively toward the station. Each node is connected to one or several nodes. The network should be designed to minimize their total power consumption for example by using a network controller node, because cooperative communication consume their power so much and sensor nodes do not always equip sufficient power supplies. As the controll node equip sufficient power supplies, the node can collect condition of the sensor nodes interactively in detail. Then the controll node calculate more suitable solution and tell the solution to sensors.

Earlier research and development of wireless sensor networks were motivated by military applications. In front line of battle field, there is not always sufficient power supplies and there is a strong demand to collect environmental data in a widespread space. The method to spread the sensor with small battery and construct wireless network was suggested. Today, the networks are likely to be used in agricultural applications. In a farm, a forest and fisheries, there is no sufficient power supply and strong demands to collect environmental information.

A size and a cost of sensor nodes is variable. The size and cost constraints on sensor nodes result in corresponding constraints on resources such as energy, memory, computational speed, and communication bandwidth. The topology of a network can vary from a simple star network to an advanced multi-hop wireless mesh network. The propagation technique used between the hops of the network can be routing or flooding.

### 1.1.3 Big Data

Big data is a term for a collection of datasets so large and complex that they become difficult to process using on-hand database management tools or traditional data processing applications<sup>6</sup>. The challenges of big data include capturing, storage, searching, sharing, transfers, analysis, and visualization. The trend toward larger datasets is due to the additional information derivable from an analysis of a single large set of related data, as compared to separate smaller sets with the same total amount of data, allowing correlations to be found for spotting business trends, determining the quality of research, preventing diseases, linking legal citations, combating crime, and determining real-time roadway traffic conditions[3].

---

<sup>6</sup><http://www.zdnet.com/blog/virtualization/what-is-big-data/1708>

At multiple terabytes in size, the text and images on Wikipedia<sup>7</sup> are a classic example of big data. As of 2012, limits on the size of datasets that are feasible to process in a reasonable amount of time were on the order of exabytes of data. Data scientists regularly encounter limitations from large datasets in many areas, including meteorology, genomics, connectomics, complex physics simulations, and biological and environmental research. The limitations also affect Internet searches, and finance and business informatics. Datasets have grown in size in part because they are increasingly being gathered by ubiquitous information-sensing mobile devices, aerial sensory technologies (remote sensing), software logs, cameras, microphones, radio-frequency identification readers, and wireless sensor networks. The world's per-capita technological capacity to store information has roughly doubled every 40 months since the 1980s; as of 2012, 2.5 quintillion ( $2.5 \times 10^{18}$ ) bytes of data are created each day[4]. The challenge for large enterprises is determining who should own the big data initiatives that straddle entire organizations.

Big data are difficult to work with using most relational database management systems, and desktop statistics and visualization packages, requiring instead massively parallel software running on tens, hundreds, or even thousands of servers[5]. What is considered “big data” varies depending on the capabilities of the organization managing the set, and on the capabilities of the applications that are traditionally used to process and analyze the dataset in its domain. For some organizations, facing hundreds of gigabytes of data for the first time may trigger a need to reconsider data management options. For others, it may take tens or hundreds of terabytes before data size becomes a significant consideration[6].

Big data usually includes datasets with sizes beyond the ability of commonly used software tools to capture, curate, manage, and process the data within a tolerable amount of time[7]. Big data sizes are constantly moving targets, and as of 2012, range from a few dozen terabytes to many petabytes of data in a single dataset. These targets move owing to constant improvements in traditional DBMS technology, as well as new databases such as NoSQL and their ability to handle larger amounts of data. With this difficulty, new platforms of big data tools are being developed to handle various aspects of large quantities of data.

In a 2001 research report and related lectures, META Group (now Gartner) analyst Doug Laney defined data growth challenges and opportunities as being three-dimensional, i.e., increasing volume (amount of data), velocity (speed of data in and out), and variety (range of data types and sources)[8]. Gartner, and now much of the industry, continue to use this 3V model for describing big data. In 2012, Gartner updated its definition

---

<sup>7</sup><http://www.wikipedia.org>

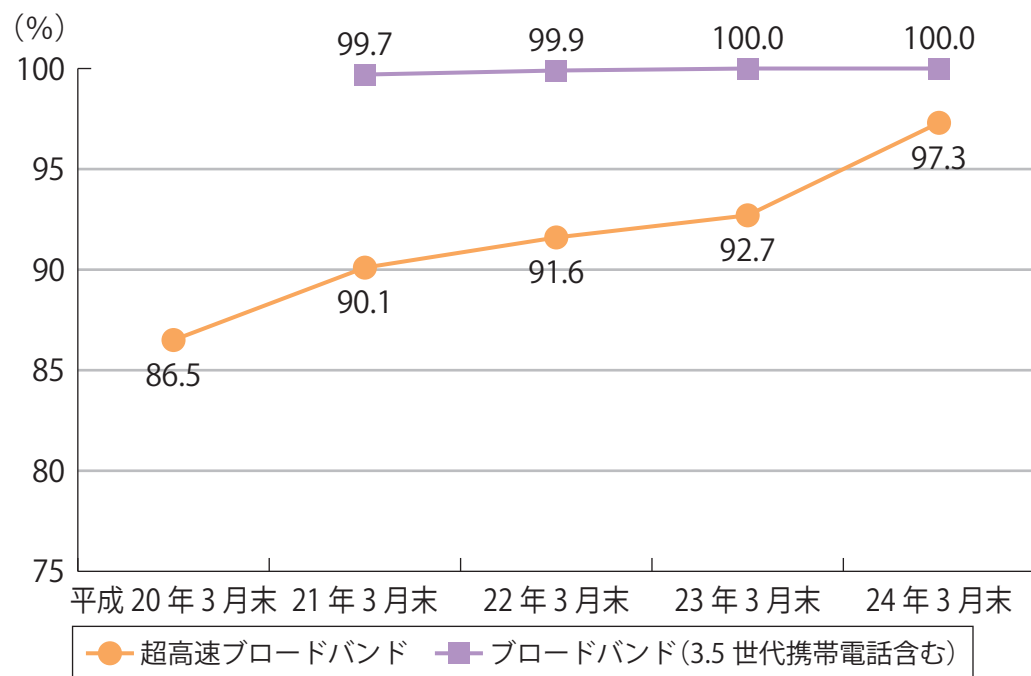


FIGURE 1.5: Changes in the maintenance conditions for the broadband connection platform in Japan (source: 平成 25 年版情報通信白書 (総務省) licensed under CC-BY 2.1 JP).

as follows; “Big data is high volume, high velocity, and/or high variety information assets that require new forms of processing to enable enhanced decision making, insight discovery, and process optimization”. Additionally, a new V, veracity, has been added by some organizations to describe this[9].

While Gartner ’ s definition (the 3Vs) is still widely used, the growing maturity of the concept has fostered a more sound difference between big data and business intelligence, regarding data and their use:

- business intelligence uses descriptive statistics from data with a high information density to measure items, detect trends, and so on;
- big data uses inductive statistics with data with a low information density whose huge volume allow the inference of laws (regressions), thereby giving (within the limits of inference reasoning) big data some predictive capabilities.

## 1.2 Multihome Network

v

Along with the development of the video image quality, the development of household broadband lines has also been an important factor of the development of Internet-based communication. In North America, household broadband access to the Internet began in 1996, when Rogers Communications introduced the first cable modem service in Canada. In the same year in Japan, Musashino Mitaka Cable Television provided a dialup connection to the Internet using a cable television modem. The price of this service was too expensive to be said to household broadband line. In August 1999, Naganoken Kyodo Densan Co., Ltd provided a dialup connection to the Internet using ADSL technology at low prices. NTT East and West, major communication companies of Japan, provided ADSL connection in December 2000, and FTTH connection in August 2001. According to a white paper on information and communications in Japan, the number of households available for ultra high-speed broadband connection to the Internet is 5,235 households with a ratio of 97.3 % and the number available for broadband connection is 5,377 households with a ratio of 100 % as of the end of March, 2012<sup>8</sup>. Ultra high-speed broadband connection includes all fiber-to-the-home (FTTH) connections. In addition, it also includes over 30 Mbps cable television (CATV) Internet connection, fixed wireless access (FWA), and broadband wireless access (BWA). Broadband connections include FTTH, a digital subscriber line (DSL), CATV Internet connection, FWA, satellite connection, BWA, and 3.5 G mobile phones (Fig. 1.5). Based on this white paper, Fig. 1.6 shows the fixed broadband spread ratio, FTT ratio, and per capita Internet spread ratio in Japan, Korea, China, Singapore, Italy, Canada, Austria, Netherlands, Finland, Swiss, Australia, France, the US, New Zealand, Portugal, the UK, Germany, Spain, Belgium, Denmark, Sweden, India, Brazil, Russia, South Africa, Norway, and Malaysia<sup>9</sup> [10].

Multihoming refers to a computer or device connected to more than one computer network<sup>10</sup>. As an example, it can be used to increase the reliability of an Internet Protocol network, such as users served by more than one Internet service provider. Multihoming can also add to the costs, including upfront costs (search, initial investment, and training costs), ongoing costs (membership fees and maintenance costs), and exit costs (the salvage value of hardware/software, and termination costs). In an IP context, there are several methods for multihoming, separate from the actual protocols used, among which the most important are single link, multiple IP address, multiple interfaces, single IP address per interface, multiple links, single IP address, and multiple links, multiple IP

<sup>8</sup><http://www.soumu.go.jp/johotsusintokei/whitepaper/h25.html>

<sup>9</sup><http://www.soumu.go.jp/johotsusintokei/whitepaper/eng/WP2012/chapter-1.pdf>

<sup>10</sup><http://en.wikipedia.org/wiki/Multihoming>

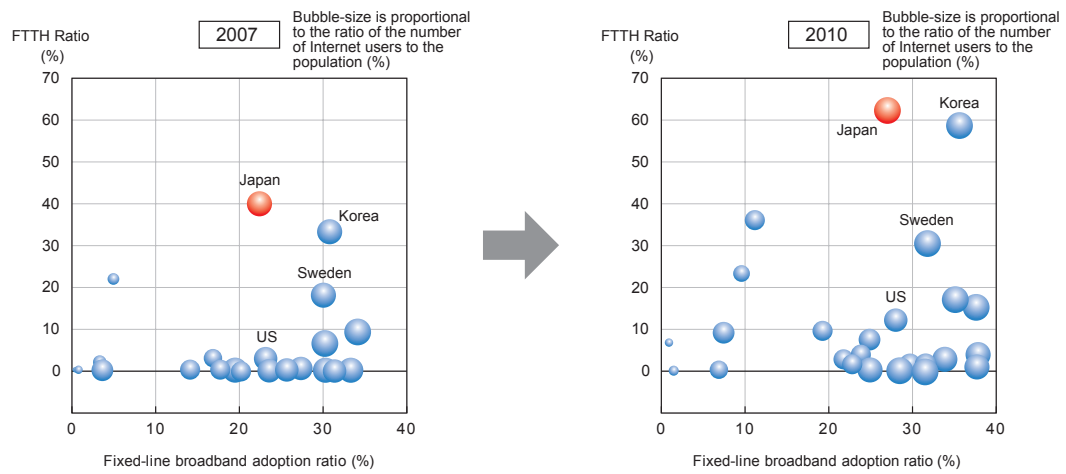


FIGURE 1.6: Fixed broadband spread ratio, FTT ratio, and per capita Internet spread ratio.

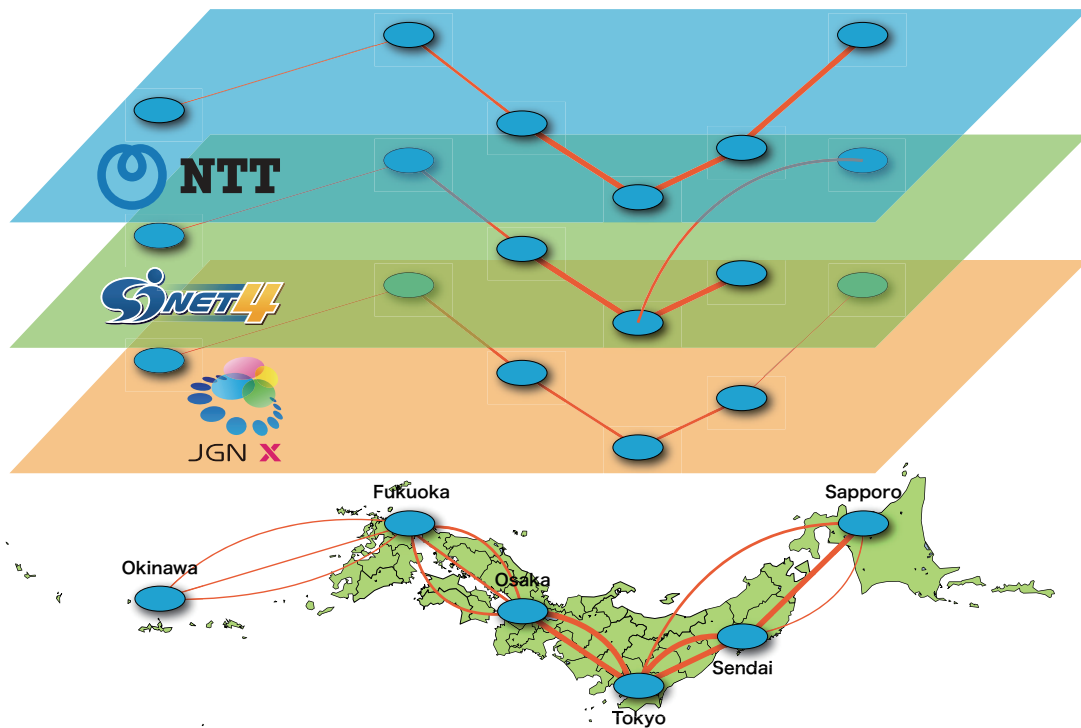


FIGURE 1.7: Diagram of multihoming.

address. While multihoming is generally used to eliminate network connectivity as a potential single point of failure (SPOF), certain implementation caveats apply that can affect the success of such a strategy.

In particular, to eliminate the network SPOF:

**Upstream connectivity** A given network-operation center must have multiple upstream links to independent providers. Furthermore, to lessen the possibility of simultaneous damage to all upstream links, the physical location of each of these upstream links should be physically diverse, i.e., far enough apart that a piece of machinery (such as a backhoe) will not accidentally sever all connections at the same time.

**Routers** Routers and switches must be positioned such that no single piece of network hardware controls all network access to a given host. In particular, it is not uncommon to see multiple Internet uplinks all converge at a single edge router. In such a configuration, the loss of that single router will disconnect the Internet uplink, despite the fact that multiple ISPs are otherwise in use.

**Host connectivity** A “reliable” host must be connected to the network over multiple network interfaces, each connected to a separate router or switch. Alternatively, and preferably, the function of a given host can be duplicated across multiple computers, each of which is connected to a different router or switch.

**Referencing Entities** Not only must a host be accessible, but in many cases it must also be “referenced” to be useful. For most servers, this means in particular that the name resolution to that server should be functional. For example, if the failure of a single element blocks users from properly resolving the DNS name of that server, then the server will be effectively inaccessible, despite its otherwise connected state.

The elimination of a single point of failure is achieved only when each component that may potentially fail is duplicated.

### 1.2.1 Overlay Networking

An overlay network is a computer network built on the top of another network. Nodes in an overlay network can be thought of as being connected by virtual or logical links, each of which corresponds to a path, perhaps through many physical links, in the underlying network. For example, distributed systems such as cloud computing, peer-to-peer networks, and client-server applications are overlay networks because their nodes run on top of the Internet. The Internet was originally built as an overlay upon a telephone network, while today (through the advent of VoIP), telephone networks are increasingly turning into an overlay network built on top of the Internet<sup>11</sup>.

---

<sup>11</sup>[http://en.wikipedia.org/wiki/Overlay\\_network](http://en.wikipedia.org/wiki/Overlay_network)

Overlay networks are used in telecommunications because of the availability of digital circuit switching equipment and optical fiber. Telecommunication transport and IP networks (which combined make up the broader Internet) are all overlaid with at least an optical layer, a transport layer, and an IP or circuit layer (in the case of PSTN). Private enterprise networks were first overlaid on telecommunication networks such as frame relay and asynchronous transfer mode packet switching infrastructures, but migration from these (now legacy) infrastructures to IP-based MPLS networks and virtual private networks has started. From a physical standpoint, overlay networks are quite complex as they combine various logical layers that are operated and built by various entities (businesses, universities, governments, etc.), but they allow the separation of concerns that over time has permitted the build-up of a broad set of services that could not have been proposed by a single telecommunication operator (e.g., broadband Internet access, voice over IP or IPTV, or competitive telecom operators)[11].

Nowadays, the Internet has become the basis for more overlaid networks that can be constructed to permit the routing of messages to destinations not specified by an IP address. For example, distributed hash tables can be used to route messages to a node having a specific logical address, whose IP address is not known in advance. Overlay networks have also been proposed as a way to improve Internet routing, such as through quality of service guarantees to achieve higher-quality streaming media. Previous proposals such as IntServ, DiffServ, and IP multicast have not seen wide acceptance largely because they require a modification of all routers in the network. On the other hand, an overlay network can be incrementally deployed on end-hosts running the overlay protocol software, without cooperation from ISPs. The overlay has no control over how packets are routed in the underlying network between two overlay nodes, but it can control, for example, the sequence of overlay nodes a message traverses before reaching its destination. For example, Akamai Technologies manages an overlay network that provides reliable and efficient content delivery (a type of multicast). Academic research includes End System Multicast and Overcast for multicast, RON (Resilient Overlay Network) for resilient routing, and OverQoS for quality of service guarantees, among others. For example, Virtela Technology Services provides an overlay network in 90 or more countries on top of more than 500 different underlying telecom providers.

### 1.3 Traffic Engineering

Traffic engineering or telecommunications traffic engineering is the application of traffic engineering theory to telecommunications. Teletraffic engineers use their knowledge of

statistics including queuing theory, the nature of traffic, their practical models, and measurements and simulations to make predictions and plan telecommunication networks such as telephone networks or the Internet. These tools and knowledge help provide reliable services at lower cost. The field was created through the work of A. K. Erlang for circuit-switched networks, but is applicable to packet-switched networks, as they both exhibit Markovian properties, and can hence be modeled by a Poisson arrival process, for example. The crucial observation in traffic engineering is that, in large systems, the law of large numbers can be used to make the aggregate properties of a system much more predictable over a long period of time than the behavior of individual parts of the system<sup>12</sup>.

The measurement of traffic in a public switched telephone network (PSTN) allows network operators to determine and maintain the quality of service (QoS), and in particular, the grade of service (GoS), that they promise their subscribers. The performance of a network depends on whether all origin-destination pairs receive satisfactory service.

Networks are handled as follows:

**loss systems** where calls that cannot be handled are given an equipment busy tone; or

**queuing systems** where calls that cannot be handled immediately are queued.

Congestion is defined as a situation in which exchanges or circuit groups are inundated with calls, and not all subscribers are able to be served. Special attention must be given to ensure that such high loss situations do not arise. To help determine the probability of congestion, operators should use Erlang formulas or an Engset calculation. Exchanges in the PSTN make use of trunking concepts to help minimize the cost of the equipment for the operator. Modern switches generally have full availability and do not make use of grading concepts. Overflow systems make use of alternative routing circuit groups or paths to transfer excess traffic, and thereby reduce the possibility of congestion.

A very important component in a PSTN is the SS7 network used to route signaling traffic. As a supporting network, it carries all the signaling messages necessary to set up, break down, or provide extra services. The signaling enables the PSTN to control the manner in which the traffic is routed from one location to another. The transmission and switching of calls are performed using the principle of time-division multiplexing (TDM). TDM allows multiple calls to be transmitted along the same physical path, reducing the infrastructure costs.

---

<sup>12</sup>[http://en.wikipedia.org/wiki/Teletraffic\\_engineering](http://en.wikipedia.org/wiki/Teletraffic_engineering)

### 1.3.1 Active Networking

An active network architecture is composed of execution environments (similar to a Unix shell that can execute active packets), a node operating system capable of supporting one or more execution environments. It also consists of active hardware, capable of routing or switching, and executing code within active packets. This differs from a traditional network architecture that seeks robustness and stability by attempting to remove complexity, and in the ability to change its fundamental operation from underlying network components. Network processors are one of the means of implementing active networking concepts. Active networks have also been implemented as overlay networks<sup>13</sup>.

Active network research addresses the nature of how best to incorporate an extremely dynamic capability within networks[12]. To do so, active network research must address the problem of optimally allocating computations versus communication within communication networks[13]. A similar problem related to the compression of code as a measure of complexity is addressed using algorithmic information theory. One of the challenges of active networking has been the inability of information theory to mathematically model an active network paradigm and enable active network engineering. This is due to the active nature of a network in which communication packets contain code that dynamically changes the operation of the network. Fundamental advances in information theory are required to understand such networks[14].

As the limit in the reduction of the transistor size has been reached with current technology, active networking concepts are being explored as a more efficient means of accomplishing computation and communication[15]. More on this can be found in nano-scale networking.

There are currently two realistic implementations and streams of active networking: a software-defined network, and OpenFlow.

### 1.3.2 Software-defined Networking

Software-defined networking (SDN) is an approach to computer networking that evolved from work conducted at UC Berkeley and Stanford University in around 2005. SDN allows network administrators to manage network services through an abstraction of lower-level functionality. This is done by decoupling the system that makes decisions about where traffic is sent (the control plane) from the underlying systems that forward traffic to the selected destination (the data plane). The inventors and vendors of these

---

<sup>13</sup>[http://en.wikipedia.org/wiki/Active\\_networking](http://en.wikipedia.org/wiki/Active_networking)

systems claim that this simplifies networking<sup>14</sup>. SDN requires a method for the control plane to communicate with the data plane. One such mechanism, OpenFlow, is often misunderstood to be equivalent to SDN, but other mechanisms can also fit into the concept. The Open Networking Foundation was founded to promote SDN, and OpenFlow has become popular as a marketing tool based on the term cloud computing<sup>15</sup>.

Internet protocol (IP) based networks were initially built based on the notion of autonomous systems (AS). This notion allows networks to scale and extend through connected junctions that forward packets to a reasonable next hop based on partial need-to-know information. This approach to networking is simple, and has proven to be resilient and scalable. The principle of an AS does not allow the designated destinations to move without changing their identity as far as the packet delivery service is concerned. The topological location of the destinations, which is the network interface they are attached to, dictates their identity. In addition, using only a basic AS, it is hard to specify other identity qualities, such as logical grouping, access control, quality of service, and intermediate network processing, or to specify aspects that relate to a sequence of packets that form a flow or networked conversation.

Complementary standards by the Internet Engineering Task Force (IETF) were put in place to augment identity-specific needs, standards such as virtual LANs, and virtual private networks, among many others. These incremental standards have increased complexity in their network element specifications and network interface configurations by network operators.

As elastic cloud architectures and dynamic resource allocation evolve, and mobile operating systems and virtual machine usage grows, the need has arisen for an additional layer of SDN. Such a layer allows network operators to specify network services, without coupling these specifications with network interfaces. This enables entities to move between interfaces without changing identities or violating specifications. It can also simplify network operations, where global definitions per identity do not have to be matched to each and every interface location. Such a layer can also reset some of the complexity build-up in network elements by decoupling the identity and flow-specific control logic from basic topology-based forwarding, bridging, and routing.

Global software-defined control also tracks specific flow contexts based on the aspects of the source and destination identities. A mechanism for driving network hardware has been added and adopted by network equipment manufacturers for the purpose of sharing edge driving between software defined edges and vendor specific bridging and routing.

---

<sup>14</sup><https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf>

<sup>15</sup>[http://en.wikipedia.org/wiki/Software-defined\\_networking](http://en.wikipedia.org/wiki/Software-defined_networking)

A set of open commands for forwarding was defined in the form of a protocol known as OpenFlow. The OpenFlow protocol enables globally aware software controllers, which may be centralized or distributed, to drive the network edge hardware to create an easily programmable identity-based overlay on top of a traditional IP core.

SDN is a step in the evolution toward programmable and active networking. SDN allows network administrators to have central programmable control of network traffic through a controller without requiring physical access to the network switches<sup>16</sup>. An SDN configuration can create a logical network control plane where the hardware is physically decoupled from the data forwarding plane, i.e., a network switch can forward packets, and a separate server can run the network control plane. The decoupling allows for the control plane to be implemented using a different distribution model than the data plane. Control plane development and runtime environment tasks can then be run on a different platform (other than the low-powered management CPUs found on hardware switches and routers).

On the other hand, Open Networking Foundation (ONF) defines SDN as follows<sup>17</sup>: SDN is an emerging architecture that is dynamic, manageable, cost-effective, and adaptable, making it ideal for the high-bandwidth, dynamic nature of today's applications. This architecture decouples the network control and forwarding functions enabling the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services. The OpenFlow® protocol is a foundational element for building SDN solutions. And SDN architecture consists of “Directly programmable”, “Agile”, “Centrally managed”, “Programmatically configured” and “Open standards-based and vendor-neutral”. Descriptions of the each term is shown below.

**Directly programmable** Network control is directly programmable because it is decoupled from forwarding functions.

**Agile** Abstracting control from forwarding lets administrators dynamically adjust network-wide traffic flow to meet changing needs.

**Centrally managed** Network intelligence is (logically) centralized in software-based SDN controllers that maintain a global view of the network, which appears to applications and policy engines as a single, logical switch.

**Programmatically configured** SDN lets network managers configure, manage, secure, and optimize network resources very quickly via dynamic, automated SDN

---

<sup>16</sup><http://www.networkworld.com/news/2012/082912-insider-sdn-262010.html>

<sup>17</sup><https://www.opennetworking.org/sdn-resources/sdn-definition>

programs, which they can write themselves because the programs do not depend on proprietary software.

**Open standards-based and vendor-neutral** When implemented through open standards, SDN simplifies network design and operation because instructions are provided by SDN controllers instead of multiple, vendor-specific devices and protocols.]

### 1.3.3 OpenFlow

OpenFlow is a communications protocol that gives access to the forwarding plane of a network switch or router over a network[16]. The Open Networking Foundation (ONF), a user-led organization dedicated to the promotion and adoption of SDN, manages the OpenFlow standard. ONF defines OpenFlow as the first standard communications interface defined between the controls and forwarding layers of the SDN architecture. OpenFlow allows direct access to and the manipulation of the forwarding plane of network devices such as switches and routers, both physical and virtual (hypervisor-based). The absence of an open interface to the forwarding plane has led to the characterization of today's networking devices as monolithic, closed, and mainframe-like. A protocol like OpenFlow is needed to move the network control out of proprietary network switches and into control software that is both open source and locally managed<sup>18</sup>.

In simpler terms, OpenFlow allows a path of network packets through a network of switches to be determined by software running on multiple routers (a minimum of two routers, primary and secondary, act as observers). This separation of the control from the forwarding allows for more sophisticated traffic management than is feasible using access control lists (ACLs) and routing protocols. Its inventors consider OpenFlow an enabler of SDN. A number of network switch and router vendors have announced their intent to support, or are shipping supported switches for, OpenFlow, including Big Switch Networks, Brocade Communications, Arista Networks, Cisco, Force10, Extreme Networks, IBM, Juniper Networks, Digisol, Larch Networks, Hewlett-Packard, NEC, and MikroTik. Some network control plane implementations use this protocol to manage the network forwarding elements. OpenFlow is mainly used between the switch and controller on a secure channel.

Traditional methods for TE can be summarized into two types [17]. One type is an offline method that measures the total amount of traffic demand in advance, and then calculates the optimal route using the maximum amount of traffic demand. It is difficult for this method to follow real-time changes in traffic demand, and is thus said to lack sensibility

---

<sup>18</sup><http://en.wikipedia.org/wiki/OpenFlow>

and adaptability. In addition, this method is highly effective when the measured traffic demand is equal to the real traffic demand. However, these two types of traffic demand are hardly completely equal.

To deal with these issues, another type of TE method, called an online method, was proposed and researched. Most of this method requires an MPLS network [18–22]. In using the MPLS TE method, it is necessary to arrange a comparatively expensive router, as some cheap routers have inter-connectivity problems [23]. Many online methods with MPLS contain the possibility of SPOF and adjustability issues on the ingress edge in a complex topology network. In addition, they require many label switched paths (LSPs) to be set up beforehand. Therefore, in a wide-area large-scale network, and under network conditions where various traffic demands and network congestion have broken out, it is hard to predict and select detour routes that avoid congestion paths by them. Other online methods without an MPLS are known to create path oscillation [24], which produces problems such as an increased packet loss and latency jitter, which decrease the transmission quality and network stability.

According to these problems and challenges for offline and online TE methods, a proposed TE method must provide the following four features, “real-time responsiveness for changes of real network traffic”, “scalability and flexibility for large scale complex topology”, “SPOF-less distributed autonomy” and “compatibility of both a proper traffic distribution and moderation to path oscillation”.

There are two methods used for network traffic distribution. One method is a deterministic approach that distributes traffic demands equally for multiple paths that are preliminarily defined. Using this method, it is difficult to change the number of paths adaptively, or change existing paths into new detour paths according to changes in the network conditions. On the other hand, for the second method, called a probabilistic approach, the traffic is distributed in a weighed random fashion toward the paths to decrease the amount of packet re-ordering caused by the latency difference among paths. It is also necessary for this method to avoid packet loops [25]. Although methods in which the routed paths of the packets are described in their header have been considered to avoid such loops, such methods may worsen the scalability because of the complicated packet process for the routing nodes. However, for general Internet use, the quality of the network applications can be maintained by the applications themselves using error correction [26].

Routing nodes can autonomously measure the utilization ratio of the communication lines and one-way delay to an adjacent routing node. Although the utilization ratio can be used to evaluate the degree of a communication line, it is known to be difficult to be converged to stable route paths in competing to accomplish lower utilization ratio. As a

result, the oscillation of the total packet latency time is increasing. On the other hand, if the total latency time is set to the target value for autonomous competition, it is easy to realize to be converged into stable route paths because the shortest latency in the weighed graph, in which the link costs are equal to the one-way delay of the link in a real network, is an optimal solution when forwarding latency can be approximated as 0 for every routing node. Though a gateway level routing method has been proposed as an adaptive TE routing method using one-way delay, this method merely avoids congestion paths, and cannot distribute traffic demands effectively [27].

We therefore propose the use of an autonomous distributed adaptive routing algorithm [28]. In the proposed algorithm, the routed path and delay time between two routing nodes are described into every packet. Using this information, every routing node can collect the one-way delay time information between itself and other adjacent nodes. Every node then makes a unique routing table using this information as adjacent node scores. Based on these scores, every node sets an evaluation value of the candidates for the next hop, and then chooses the next hop in a weighted random distribution of each value. Through network simulation evaluations, we show that the proposed algorithm can distribute more traffic demands to an entire network, and can avoid network congestion to utilize network resources. Though this algorithm has an advantage in topologies that have more detour paths, two critical problems remain for implementation. One problem is that the paths and delay information are described in all packets, and the other is the fact that delay time symmetry must be supposed for all links.

This paper proposes the use of the Network adaptive Routing algorithm for Environmental Intelligence (NREI), which is based on a distributed autonomous routing algorithm that uses a one-way delay. This algorithm can respond to network changes in real time because the routing table is updated within a short cycle. In addition, the algorithm treats packet losses as a big penalty delay to avoid unreachable nodes. This research proposes a TE method for an overlay network with the use of network applications that are not latency sensitive, and thus the network applications should recover the packet re-ordering and packet loops on their own. Latency sensitive network applications should get a handle on other overlay networks [29, 30]. The total amount of measured packet traffic is so sufficiently small that we can neglect the effect on the quality of a network transmission. Using the algorithm, we can distribute the traffic demands to detour paths around traffic congestion without the need for any manually explicit handling. NREI was implemented on PC routers and tested on an IP network for a comparison with a static routing algorithm. The test results show that the proposed routing algorithm has better adaptability for congested path avoidance and network load balancing.

The amount of network utilization for delivering high-definition (HD) movie data among wide-area facilities is on the increase, such as in the remote manipulation of industrial applications, remote medical diagnoses and operations, and real-time interaction with art works. Since the early days of IT, movie data have been delivered on computer networks for cosmic and astronomical events such as eclipses, and for regional cultural events such as festivals. The traffic demand for delivering an HD movie takes up network bandwidth and decreases the communication quality of other network applications. It is therefore necessary for certain institutes that have more than one communication line to distribute their traffic demands toward these lines using an adequate distribution ratio.

Communication lines that are connected to the institutes are also used by other applications. It is necessary to measure the amount of traffic demand for HD movie data and the available bandwidth together. At certain facilities, network use may be restricted by the network policy of the organization, such as a bandwidth cap or protocol limitation. When a large amount of traffic attempts to go through a line, exceeding its bandwidth, the quality of the delivered movie may suffer under the influence of cross traffic. It is difficult to avoid this deterioration in quality using only the local network information of each facility. In an existing operating method of movie delivery, before a large traffic demand arises, network operators derive a combination of paths that does not bring about such deterioration, and at certain facilities the routing tables are changed in the routers. The cost of this type of operation is a strong contributor to the hesitance in using computer networks for movie delivery.

To promote the use of movie delivery on a computer network, it is desirable for each facility to utilize communication lines without regard to derive and change network combination of paths by operators. Distributing traffic to certain lines to better utilize network resources is called traffic engineering (TE) [31]. Though various TE methods have been researched and developed, it is necessary for modern real-time TE intended to deliver greater than HD quality movies to not only distribute traffic demands statically, but to also change the combination of paths autonomously according to the network environment. By reducing the packet loss rate and total latency time, which have an influence on movie quality, modern TE methods have to control various traffic demands delicately.

## 1.4 Objective

An increase of the number of routing nodes. Routing tables used to route IP packets will continue to grow if we continue to rely on conventional routing algorithms. Conventional routing protocols either provide a single path between each source-destination pair, or

multiple paths of equal length. These algorithms and protocols deterministically select the best routes according to a predefined metric. Network traffic congestion is caused by a concentration of a large amount of network traffic at the routing nodes or on the communication lines. Congestion and load imbalance have therefore become more serious issues. New routing algorithms and protocols are demanded for high performance, loop-free conditions, multiple paths, traffic load-balancing, the handling of congestion, and minimizing delays [32].

Congestion also causes an increase in network delay and packet losses. Demand for packet retransmission increases the total amount of traffic, worsening congestion and negatively influencing other neighboring routing nodes and communication lines. It is one of the good solution expecting its cost to handle with enforcing network environments such as high performance network switches and large bandwidth communication lines. To avoid congestion and maintain the quality of the network lines, various TE technologies have been researched and developed beginning in the era of ATM networking [33, 34]. These technologies, such as RSVP-TE, utilize the processing ability of a routing node, and increase the network communication performance [35].

In this thesis, an adaptive routing algorithm called Routing for Environmental Intelligence (REI) is proposed, in which every network node works as a distributed autonomous agent for network routing. This approach uses algorithms from Ant Colony Optimization [36, 37] and Swarm Intelligence [38] as references. AntNet is an adaptive and distributed mobile-agent based algorithm inspired from observations of the foraging behavior of ant colonies, and introduced the ant colony optimization (ACO) metaheuristic. AntNet has shown both very good performances and robustness [39]. For much accustoming to network routing, Bonabeau and his colleagues supplemented AntNet with the ability to perform more computations at the switching nodes. They showed significant improvements in network relaxation and the response to perturbations [40]. Each node may have some feasible paths for the packets to reach their final destination. In this proposed approach, the scores of these paths are calculated based on the delay time given in the inbound data packets.

As another ant colony and swarm intelligence approach, a load balancing routing method using a genetic algorithm (GA) based on a link load metric was proposed [41, 42]. This method generates alternative routes using genetic operators to balance loads among them on a packet-switched network based on source routing, and prevents congestion as to network load condition. Though an approach using a genetic algorithm, a source routing option, and IPv4 was also proposed, it is difficult for this approach to make a detour path because many IP routers neglect the source routing option for security reasons. This approach also distributes inbound packets to the candidate next nodes

based on a stochastic rule. The distribution ratio is determined based on the feedback latency time of the packets reaching their destination router. Any paths from the source router toward the destination are regarded as “individuals” (creatures or phenotypes) on genetic algorithm. By modifying a genome (recombining and possibly randomly mutating), more adequate detour paths are expected to be generated. On the minus side of this method, while source routing algorithms are conceptually simple, they suffer from scalability and security problems. Hierarchical routing has been used to cope with the scalability problems of source routing in large internetworks. Genetic algorithms and neural networks may prove to be efficient for traffic control in high-speed networks [43]. Using fuzzy logic, an integrated CAC and routing strategy with cooperative agents was proposed [44]. If we can specify the sequence of nodes as a string assigned a score based on the delay time, stochastic genetic operations such as a cross-over or mutation are useful when searching for an optimal solution. However, source routing is not suited for the Internet because IP packet routing was originally based on a distributed routing scheme. In their paper, a distributed routing algorithm is introduced as a more scalable approach, but a defect in the packet loops exists, which makes the routing fail. The approach proposed herein is not based on a source routing algorithm in which the sequence of nodes to be visited is fully determined at the source node.

For wireless networks, a loop-free single-path distributed routing algorithm has been proposed [45]. This algorithm floods packets and determines a single-path toward the destination without a loop. The authors proposed several methods in which the node selects the best candidate node among its near neighbors according to the corresponding criterion, and forwards to the best among the joint neighbors. They also proposed flooding methods using localized algorithms to guarantee message delivery. In wired networks, a flooding approach does not fit with the utilization of network resources such as the available bandwidth. The adaptive routing algorithm described herein works in a fully distributed manner. At every node, we arrange an autonomous agent for adaptive routing. According to the path scores obtained, each agent independently chooses the best adjacent node (next hop) toward the final destination specified in the packet. To escape from local minimum solutions, we also make use of a probabilistic selection of adjacent nodes.

We implemented a network simulator to evaluate our approach. The experimental results show that REI is highly adaptable to environmental changes in network traffic conditions.

## Chapter 2

# Decentralized Stochastic Approach

### 2.1 Introduction

When network congestion or a network disturbance occurs along any route, downstream nodes may experience an improper upstream direction because the total delay time of the packets on a congested route or disturbed path must increase. As a consequence on the contraries, upstream nodes can avoid congestion direction using the reverse of this improper direction information (Figure 2.1).

With REI, any packet traveling in a network has to include two additional data fields in its header. One is for describing a path (list of node IDs) through which the packet

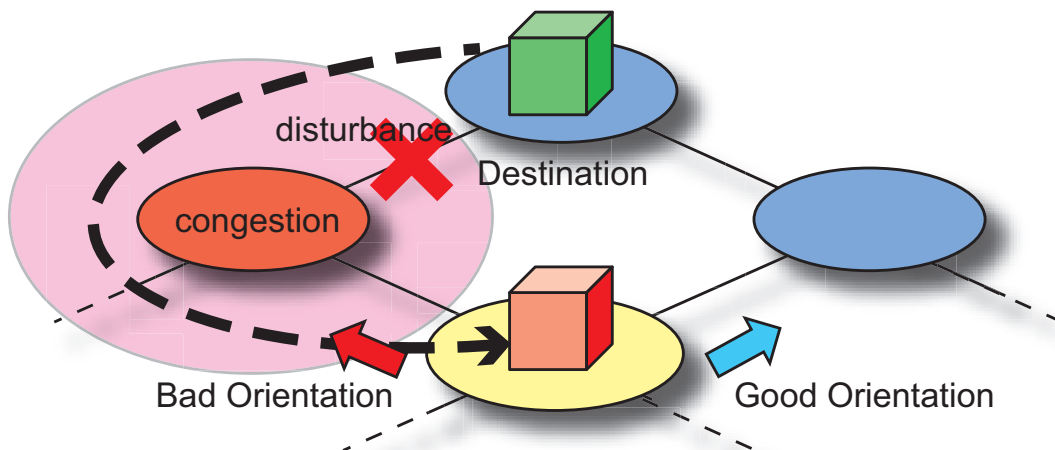


FIGURE 2.1: Congestion avoidance.

has traveled thus far. The other is for describing the latency (delay time) information between every adjacent node along the path. This latency is not determined statically, but by measuring the delay time that each packet goes through. To measure the latency more accurately, a method that estimates the NTP offset value more definitely was suggested [46].

This method reduces the estimation errors when the forward and backward paths have different bandwidths. It can estimate a clock offset and skew from one-way delay measurements with variable size packets in both directions. Its model-based accuracy can be analyzed using the assumption of small queuing delays. In this paper, we suppose that a sufficient accuracy of latency measurement is confirmed, and we do not refer to its implementation.

## 2.2 Approach by Distributed Routing Agent

### 2.2.1 Network latency

We suppose a model network for routing as a symmetric network in which all adjacent nodes can transmit and receive data packets at equal latency. The dominant factor of network latency is the process latency caused by overloaded packets at the routing nodes. Hence, we use the latency information from the inbound packets for the next node selection.

For instance, when a packet reaches the node  $N_p$ , the additional data are  $\langle N_1, d_1, N_2, d_2, \dots, N_{p-1}, d_{p-1}, N_p \rangle$ .  $N_1$  is the source node of this packet, and  $d_m$  is the latency between nodes  $N_m \sim N_{m+1}$ .

Herein, we suppose that the system clocks of all nodes are globally synchronous. The latency can be measured exactly using a time stamp put into a packet. Any node along the path can obtain the latency information from the packets. Hence, every node along the path knows how much the next node influences the total latency from here to the final destination.

### 2.2.2 Routing tables

The routing table includes a sequence of scores for each adjacent node (Figure 2.2). A score in a routing table indicates an observed latency specified in an inbound packet. Score values are therefore separately given for every different destination, such as  $L_{N_d N_a} =$

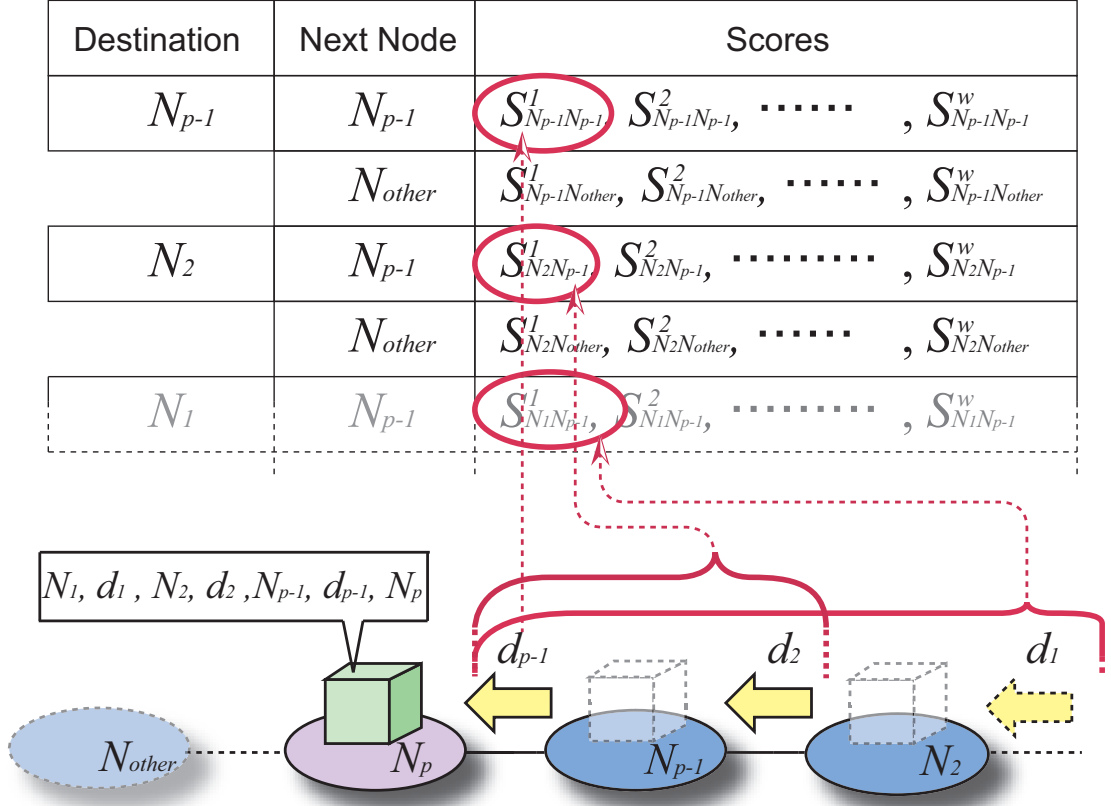


FIGURE 2.2: A REI routing table with scores.

$\langle S_{N_d N_a}^1, S_{N_d N_a}^2, \dots, S_{N_d N_a}^w \rangle$  for the destination node,  $N_d$ , with next node candidate,  $N_a$ . In addition,  $w$  is the size of the time window.

When a packet arrives at a node, the node appends the latency value calculated through equation (2.1) to the head of the score sequence associated with the next node candidate and the destination node.

$$S_{N_m N_{p-1}}^1 = \sum_{i=m}^{p-1} d_i \quad (2.1)$$

The length of the score sequence is limited, and is referred to as a *time window*. As a new score arrives, the oldest one leaves.

### 2.2.3 Next hop selection

A routing agent is assigned to each node, and makes an autonomous decision for adaptive routing. The routing procedure for forwarding data packets is as follows. When a packet arrives at a node, the agent refers to its own routing table and finds the feasible next

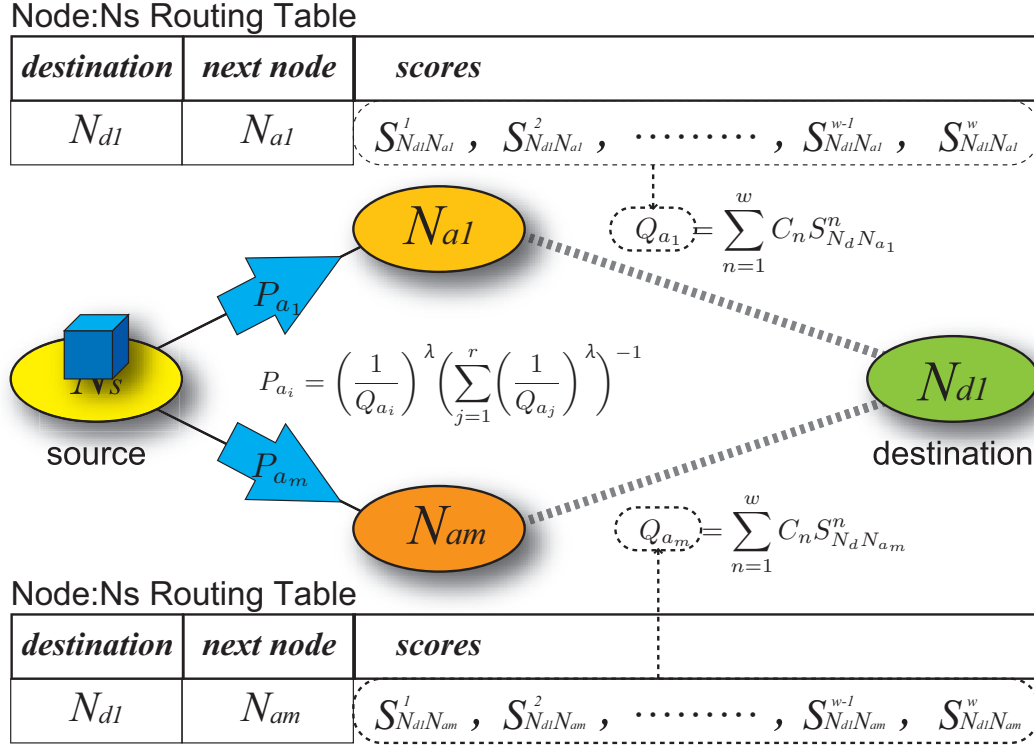


FIGURE 2.3: Probabilistic selection of the next node.

node candidates  $N_{a_i}$  which lead to the packet destination. The range of  $i$  in  $N_{a_i}$  is  $1 \leq i \leq r$ , where  $r$  is the total number of candidates.

The agent reads out all score sequences of each candidate, and calculates the weighted average latency,  $Q_{a_i} = \sum_{m=1}^w C_m S_{N_{dl}N_{a_i}}^m$ , as a representative value.  $C_m$  is a weight function. The next hop node is determined based on the probability shown in equation (2.2).

$$P_{a_i} = \left( \frac{1}{Q_{a_i}} \right)^\lambda \left( \sum_{j=1}^r \left( \frac{1}{Q_{a_j}} \right)^\lambda \right)^{-1} \quad (2.2)$$

Parameter  $\lambda$  indicates the degree of randomness in the selection. When  $\lambda \rightarrow \infty$ , the candidate which has minimum value of  $Q_{a_i}$  is selected deterministically. When  $\lambda = 0$ , all values of  $P_{a_i}$  are equivalent. In this paper we refer to a routing algorithm with a constant  $\lambda$  as REIsx.

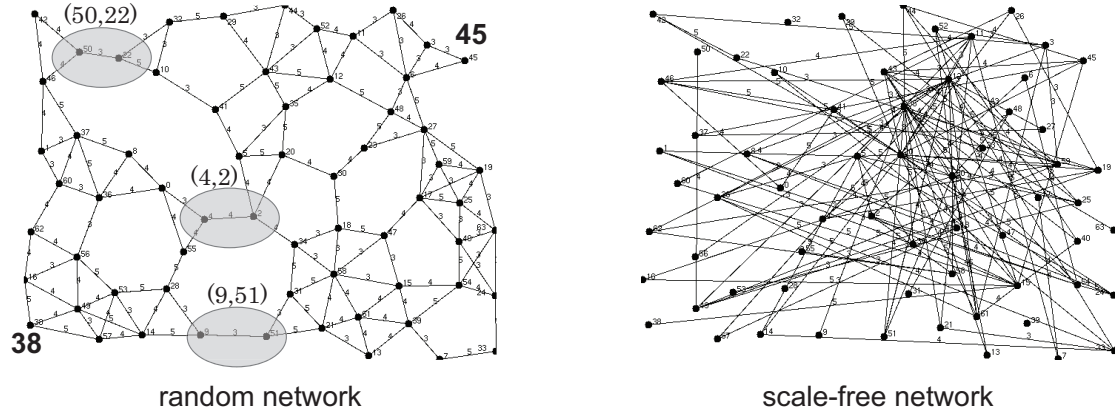


FIGURE 2.4: Network models for the simulation experiments.

## 2.3 Experimental Evaluation by Simulations

### 2.3.1 Comparative approaches

To evaluate the REIsx, we compared it to a conventional OSPF and adaptability-enhanced OSPFs (OSPF+ and OSPF++).

OSPF sends out Hello packets to establish adjacencies with neighboring nodes. In OSPF+, the Hello-interval time is 100 simulation steps (ss), and no response for 400 ss is considered to be a link down. Then the routing nodes then change the network metric table, carry out the SPF algorithm, and send out link state update packets to update the routing table.

OSPF++ also sends out Hello packets with 100 ss intervals. In addition, the latency time between the neighboring nodes is measured by using the replied Hello packets. If the latency grows to ten-times the default latency, the nodes change the network metric table and send out link state update packets.

### 2.3.2 Experimental settings

We used a random network model and a scalefree network model [47] for the simulation evaluations (Figure 2.4). Both networks include 64 routing nodes. The average network link cost (link latency) is 4 ss. Every routing node has the capability of packet processing limited to 100 packets/ss. Packets beyond this limitation are carried over to the next simulation step. At a routing node with many queued packets, the average visiting time of the packets easily becomes longer. Under these simulation conditions, one simulation step corresponds to 1 ms. The scale-free network was generated using the BA model.

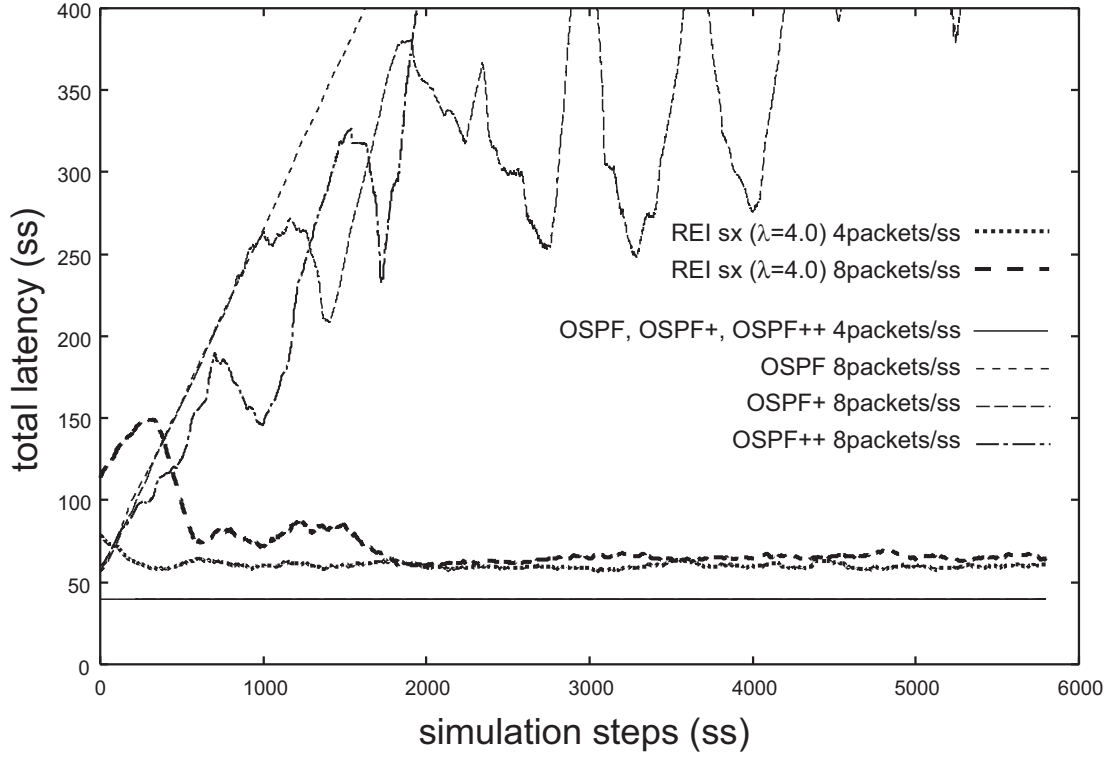


FIGURE 2.5: Average latency on the uniform traffic condition.

The Probability of link generation in each node combination for the BA model is shown in equation (2.3). In this paper, we generate a scale-free network in with an initial number of nodes,  $m_0$ , of 2 and a power index of number for the number of links  $\gamma \simeq 2.8$ .

$$\Pi(k_i) = \frac{k_i}{\sum_j k_j} \quad (2.3)$$

### 2.3.3 Uniform traffic condition

First, consider a situation in which we have widely spreading network connectivity and a heavy use of network resources. Under such a situation, various network users send and receive data packets to and from various destinations. This situation can be represented using a uniform traffic model in which each node issues the average number of packets to the destinations with a uniform distribution.

Each node issues a certain amount of packets in every step to its random destination (background traffic). Here, we focus on the latency between the two specified nodes (nodes 38 and 45). These two nodes send packets to each other at a rate of 4 packets/ss. The background traffic is changed from 2 to 16 packets/ss.

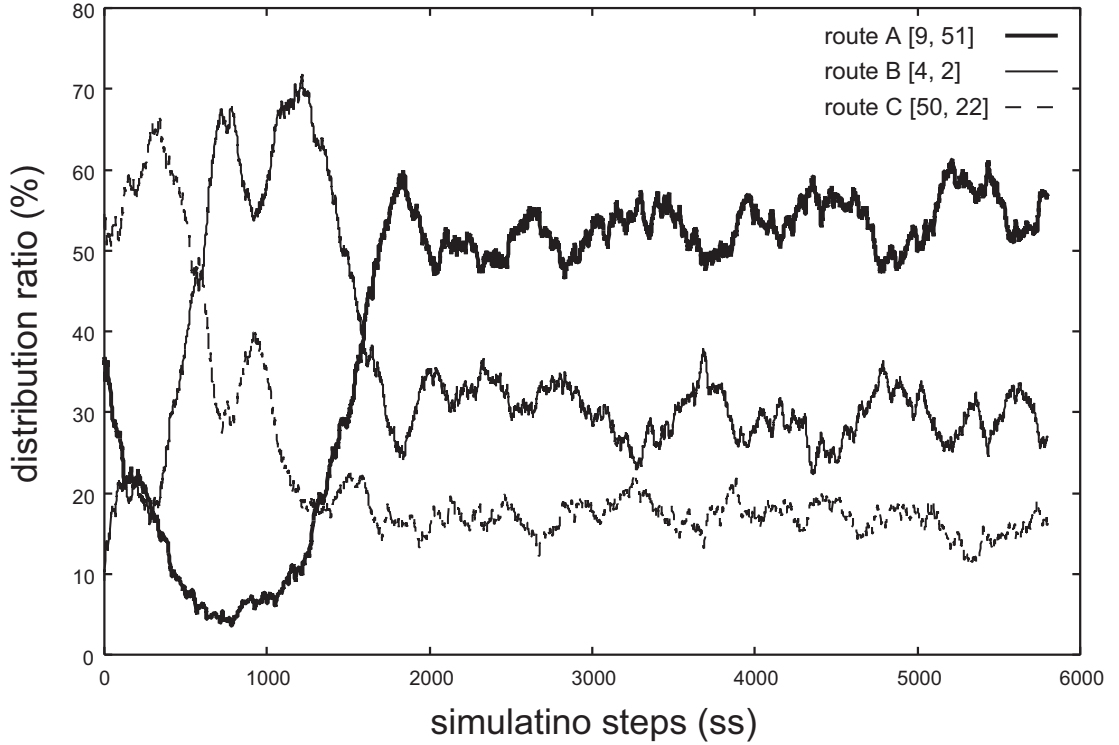


FIGURE 2.6: Route distribution ratio of REIsx.

Figure 2.5 shows the average latency compared to OSPFs under conditions in which the background traffic is 4 or 8 packets/ss. In the case of 4 packets/ss, there is no network congestion, and OSPFs show a lower average latency than REIsx. On the other hand, in the case of 8 packets/ss, network congestion occurs in some of the routing nodes. OSPFs cannot stop increasing the latency, but REIsx adaptively distributes packets over the possible paths. Hence, using REIsx, we have no fatal congestion and no increase in latency.

In the network shown in Figure 2.4, all packets from node 38 to node 45 go through one of three links: (50, 22), (4, 2), and (9, 51). Figure 2.6 shows the route distribution ratio of REIsx under a background of 8 packets/ss. The shortest latency path derived using Dijkstra's algorithm contains link (9, 51). REIsx also selects this link with the highest distribution ratio.

Figure 2.7 shows the relation between the background traffic and gradient of latency. This gradient indicates a regression coefficient calculated using the least squares method for the average total latency over the range of 0 to 6000 ss. In the case of OSPF, the total latency begins increasing at the point at which the background traffic exceeds 6 packets/ss. On the other hand, such a critical point of latency divergence for REIsx is 9 packets/ss. We can say that REIsx accepts at least 50% more traffic throughout the whole network.

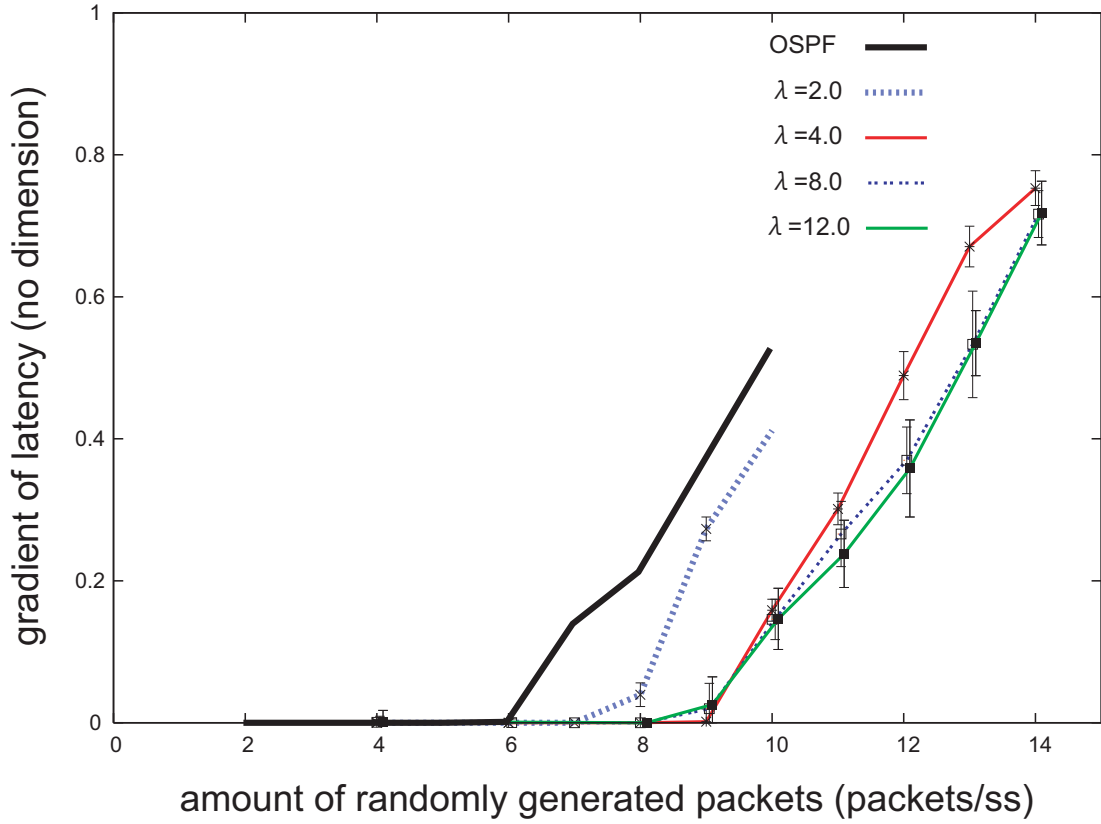


FIGURE 2.7: Background traffic and gradient of latency.

### 2.3.4 Concentrated traffic condition

We next focus on a situation of concentrated traffic between two nodes. This situation is often caused by broadband-interactive applications such as bilateral video streaming.

Under concentrated traffic conditions, the amount of background traffic is constant (4 packets/ss), and concentrated traffic (between nodes 38 and 45) is changed from 4 to 100 packets/ss. A random network is assumed as the network topology. Figure 2.8 shows the relation between the amount of concentrated traffic and the gradient of average total latency. Both OSPF and OSPF+ have less than 20 packets/ss. The gradient of latency on OSPF++ is also increased at a rate of 20 packets/ss. However, the gradient on OSPF++ is less than that on OSPF and OSPF++. On OSPF+, the gradient is reduced by 20 to 30 packets/ss. Changes in gradients from 20 to 30 packets/ss show that OSPF+ has better adaptability for concentrated traffic in a random network. On the other hand, REI shows an inudant adaptability for OSPFs. REI accepts 64 packets/ss in concentrated traffic. This is equal to a four-fold larger capacity of concentrated network traffic. In addition, under these evaluation conditions, REIsx has a great advantage for avoiding congestion and achieving load balancing.

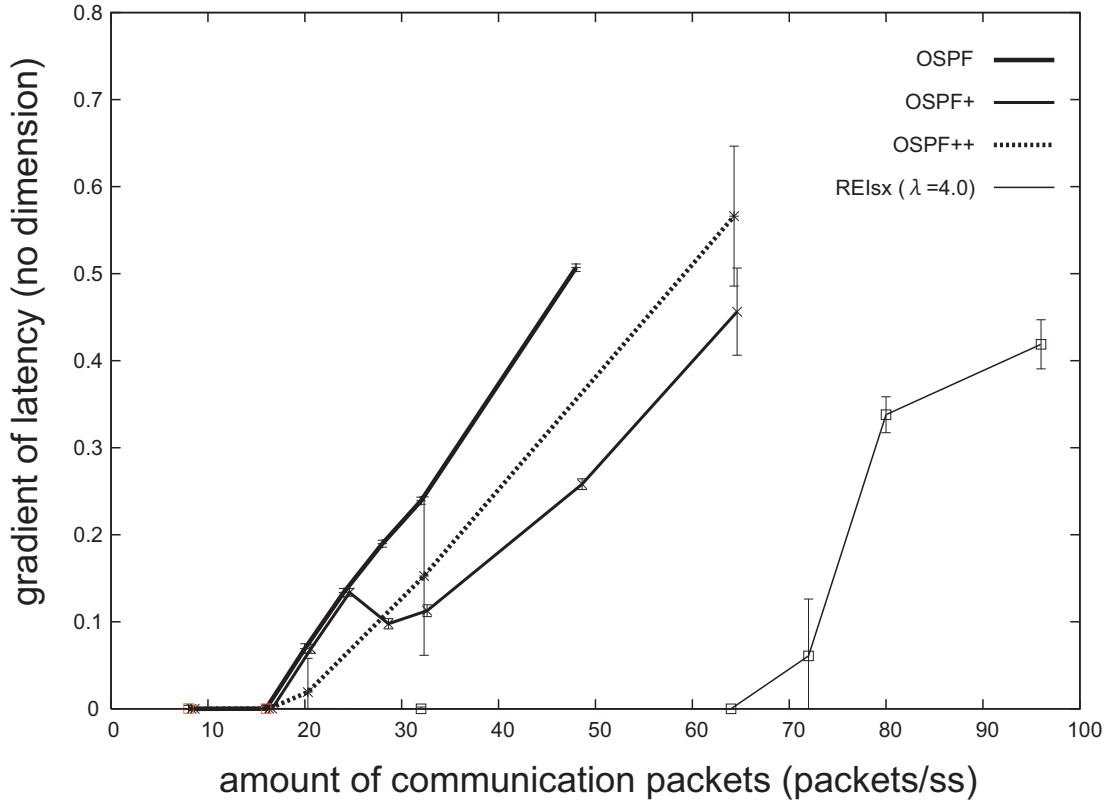


FIGURE 2.8: Concentrated traffic and gradient of latency.

### 2.3.5 Network disturbance condition

Here, we consider network disturbance caused by the contingency breakdown of links and/or nodes. It is very important for an adaptive routing algorithm to rapidly avoid problems and reroute traffic.

Under moderately concentrated traffic conditions, the cost of link (9, 51) is increased from 4 to 300 immediately at 2000 ss. At 4000 ss, the link cost is then returned to the original value of 4. Figure 2.9 shows a change in the average total latency between node 38 and 45, and Figure 2.10 shows the route distribution ratio of REIsx. The background traffic rate is 4 packets/ss, and the concentrated traffic rate is 16 packets/ss.

In the case of OSPF+ or OSPF++, routing packets are concentrated at either path (22, 50) or (4, 2) owing to the cost increase of (9, 51). This concentration brings about congestion and an increase in latency of the two paths alternately. REIsx, on the other hand, selects both paths in an autonomous and distributed manner. Hence, we have no excessive packet concentration.

Both OSPF and OSPF+ have less than 20 packets/ss. The gradient of latency on OSPF++ is also increased at 20 packets/ss. However, the gradient on OSPF++ is less

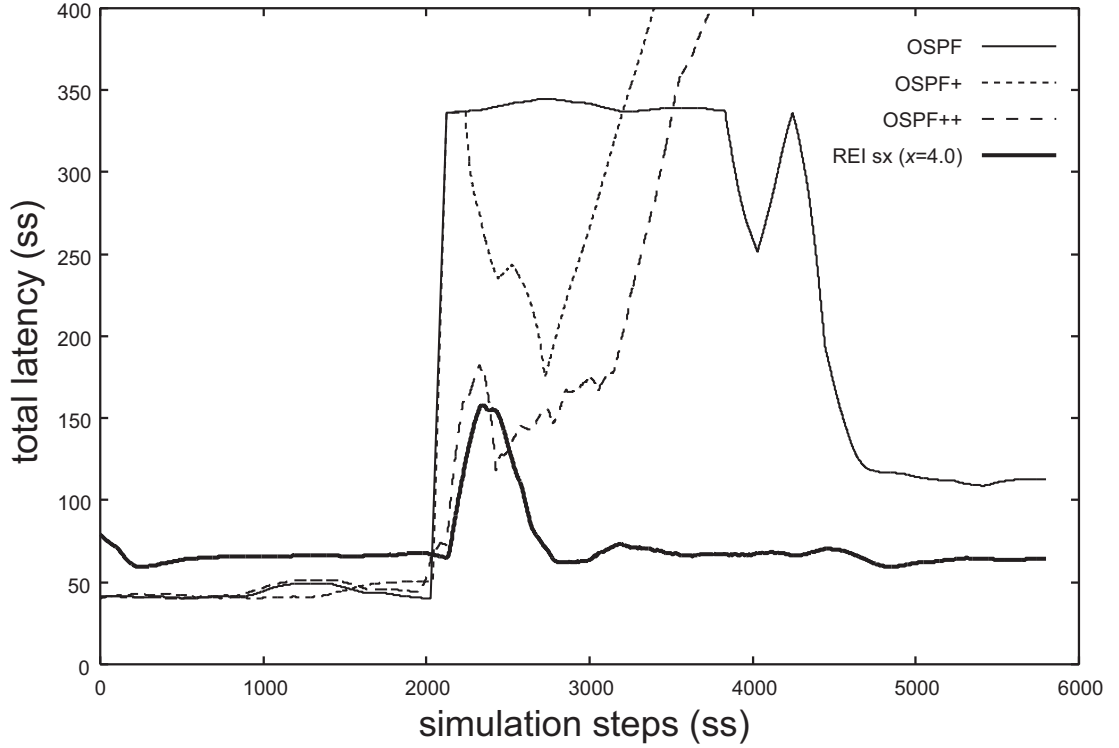


FIGURE 2.9: Average latency on the traffic disturbance.

than that on OSPF and OSPF++. On OSPF+, the gradient is reduced between 20 to 30 packets/ss. Changes in gradients from 20 to 30 packets/ss show that OSPF+ has better adaptability for concentrated traffic in a random network. On the other hand, REI shows an inudant adaptability for OSPFs. REI accepts 64 packets concentrated traffic per simulation step. This is equal to a four-fold larger capacity of concentrated network traffic. Both OSPF and OSPF+ have less than 20 packets/ss. The gradient of latency on OSPF++ is also increased at 20 packets/ss. However, the gradient on OSPF++ is less than that on OSPF and OSPF++. On OSPF+, the gradient is reduced between 20 to 30 packets/ss. Changes in gradients from 20 to 30 packets/ss show that OSPF+ has better adaptability for concentrated traffic in a random network. On the other hand, REI shows an inudant adaptability for OSPFs. REI accepts 64 packets concentrated traffic per simulation step. This is equal to a four-fold larger capacity of concentrated network traffic.

### 2.3.6 Effectiveness of parameter $\lambda$

The distribution of the selection probability for the next node is determined by parameter  $\lambda$  in REIsx. When the value of  $\lambda$  is relatively large, the total latency increasingly reaches the local minimum solution, particularly when recovering from disturbances. Though

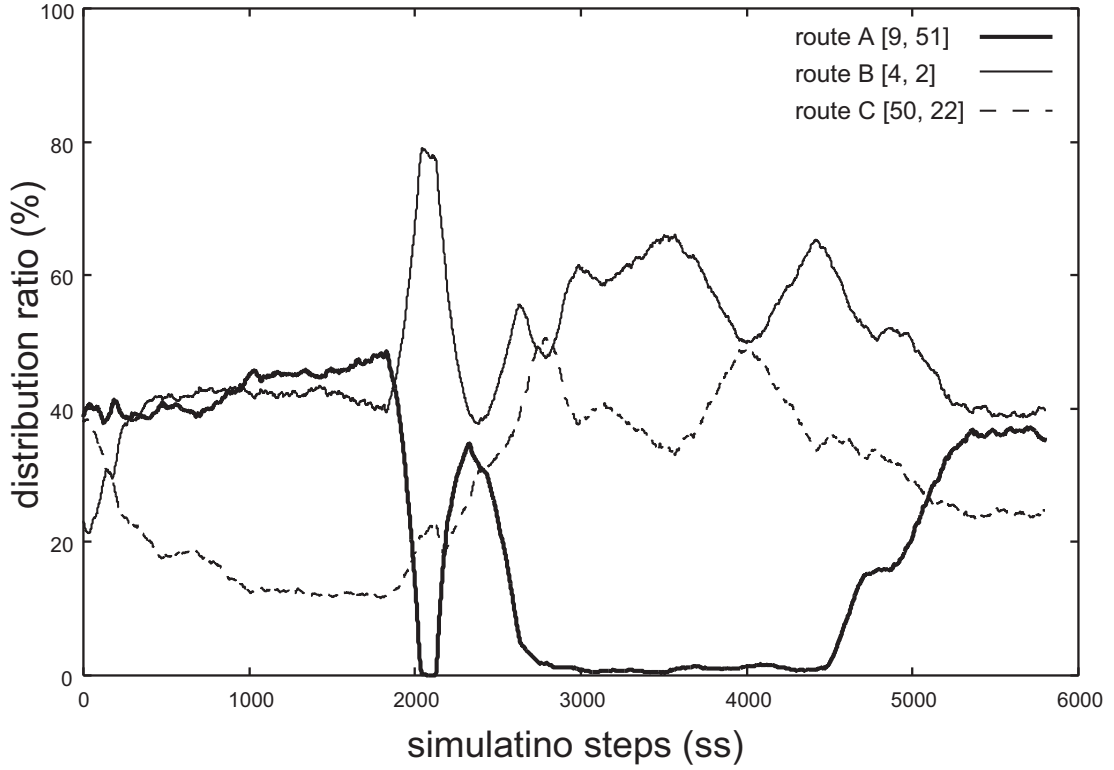


FIGURE 2.10: Route distribution of REIsx on the traffic disturbance.

TABLE 2.1:  $\lambda$  effect on the latency distribution.

	$\lambda = 1$	$\lambda = 2$	$\lambda = 3$	$\lambda = 4$	$\lambda = 5$	$\lambda = 6$	$\lambda = 8$	$\lambda = 12$
average	93.49	81.17	72.80	66.25	61.06	59.28	56.84	57.43
max	95.95	83.96	74.92	68.98	71.66	91.24	82.83	98.56
min	90.86	78.82	70.02	62.78	53.70	46.79	46.17	41.70
avg. SD	0.67	0.59	0.61	0.74	1.45	3.90	3.47	4.62
max SD	0.72	0.72	0.72	0.82	1.96	5.50	6.33	10.71
min SD	0.56	0.54	0.55	0.65	1.13	1.14	1.64	0.51

the score sequence has  $w$  scores in every node, the score of the shorter latency path is more frequently updated, and the score of the longer latency path is rarely updated when  $\lambda$  is larger. For this reason, the time needed to achieve a large score, leading to a disappearance of network disturbance, is increased. This therefore causes a difficulty in escaping from a local minimum solution.

For different  $\lambda$  values of 1 to 16, we evaluated the same network disturbance and observed an average total latency between nodes 38 and 45 in a random network of 6000 ss to 20000 ss. Table 2.1 shows the results, average total latency, maximum total latency, minimum total latency, average standard deviation (SD), maximum SD, and minimum SD. The number of packets for each random destination node is 2 packets/ss, whereas the number of packets between nodes 38 and 45 is 8 packets/ss

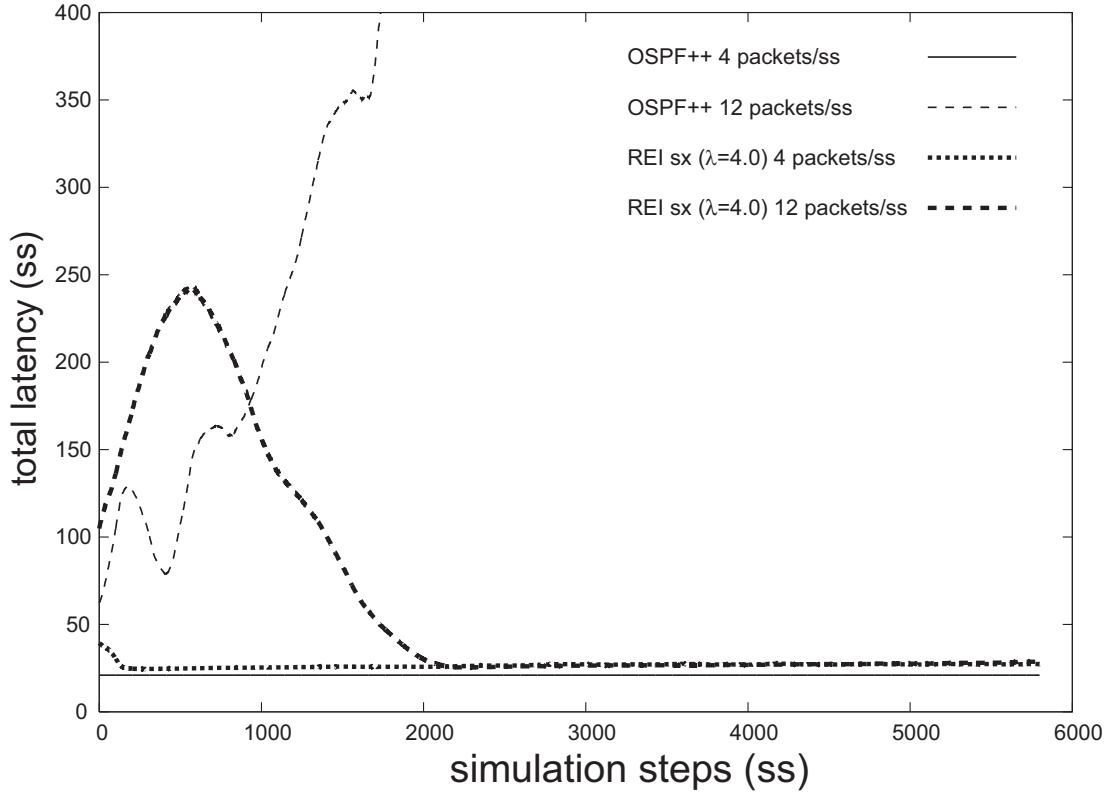


FIGURE 2.11: Average latency in SFN.

The average total latency decreases with the growing value of  $\lambda$  because the next node candidate that has the shortest latency time for the destination node is selected at a high ratio. On the other hand, this bias of distribution probability leads to a decrease in the amount of packets that travel through a longer latency path. This increases the frequency of the score updates of the shortest path and decreases that of the other paths. This delays the adaptability for a dynamic change in network traffic, and makes it easy to reach the local minimum solution of the total latency time. This appears with an increase in the maximum latency, and a standard deviation with the growing value of  $\lambda$ .

In terms of average latency, though the best solution is achieved with  $\lambda = 8$ , the jitter, which indicates the standard deviation, increases. When the amount of jitter is large, the transmission quality decreases, especially for a high-quality movie transmission. If we suppose the use of various applications, the result for  $\lambda = 4$  is the best solution because the worst value is the smallest.

To evaluate an adequate value of  $\lambda$  during actual network operation, the method for searching the optimum  $\lambda$  using the heuristic while generating the evaluation traffic is considered to be effective.

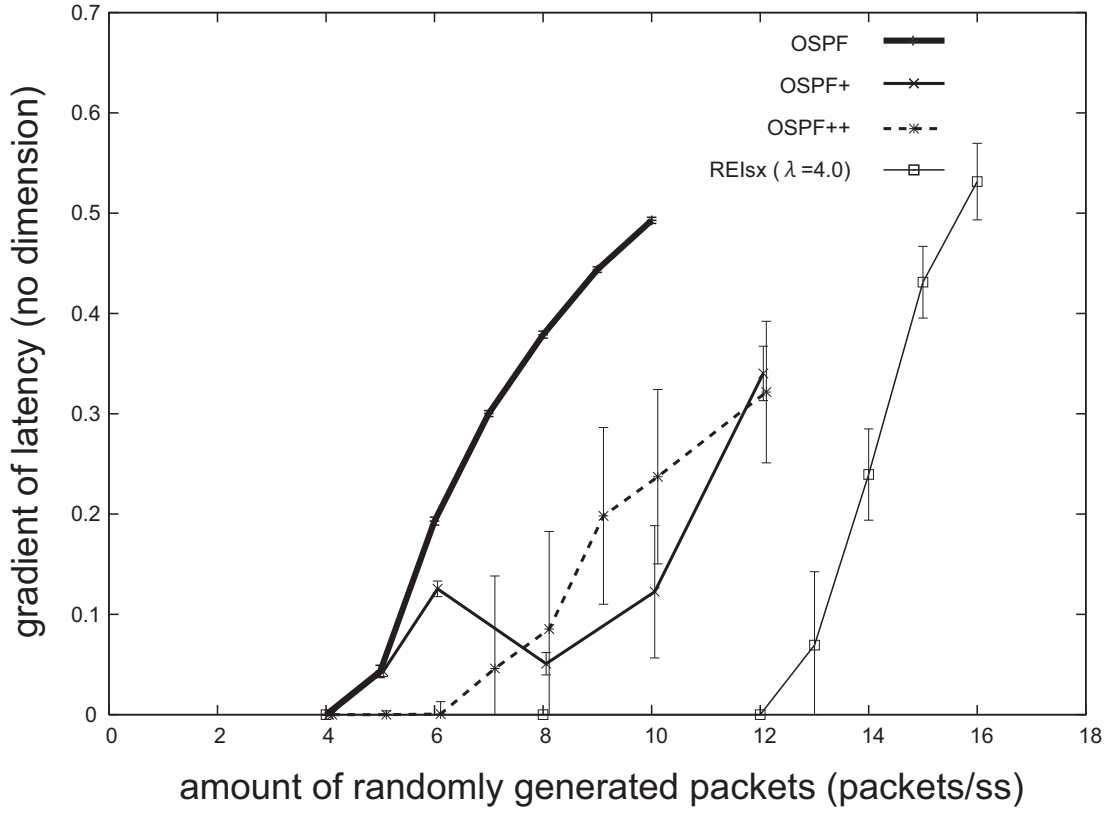


FIGURE 2.12: Gradient of average latency in SFN.

### 2.3.7 Evaluation on a scale-free network

In a scale-free network, the maximum value of the minimum hop in every combination of two nodes is smaller than in a random network. A lot of traffic travels through the core nodes, which have more links than other nodes. This means the packets converge at these core nodes.

Figure 2.11 shows the number of packets generated and the changes in total latency under uniform traffic conditions in OSPF++ and REIsx ( $\lambda = 4.0$ ). For the given topology, every node generates packets toward a random destination, and nodes 55 and 60, in which the maximum value of the minimum hop is the biggest, generate 4 packets/ss to each other. In OSPF++, the total latency monotonically increases when the background traffic is 8 packets/ss. In REIsx, the total latency converges to the shortest latency when the background traffic is 12 packets/ss. Under a smaller amount of traffic, for the same random network evaluation, OSPF++ shows a 30 % lower average latency than REIsx because of the random path selection of REIsx. Figure 2.12 shows the number of packets generated, and the gradient of the total latency.

As in the results of a random network, OSPF easily leads a packet convergence at the core nodes, and the average total latency increases for a relatively smaller number of packets generated. REIsx increases the capacity of the number of packets generated compared with OSPF+ and OSPF++, which create a new congestion by generating a new route. While OPSF++ increases the total latency at 7 packets/ss, REIsx can be tolerant until 12 packets/ss. This indicates a 70 % increase in performance.

### 2.3.8 Load balancing for network resources

REIsx uses network resources more efficiently than OSPFs. It is considered from the number of packets queued for routing at every node. From figure 2.13 to figure 2.19, the figures illustrate histograms showing the changes of time and the number of queued packets for routing at every node. In figure 2.13, 2.15, 2.17, and 2.19, the arriving packets are accumulated for 100ss, and the average number is calculated and plotted as the histogram per 1 ss from 0 packet to 1,000 packets at every 25 packets. Figure 2.13 and 2.14 show the histogram with OSPF, Figure 2.15 and 2.16 show the histogram with OSPF+, and Figure 2.17 and 2.18 show the histogram with OSPF++ on the random network. Figure 2.14, 2.16, and 2.18 shows the histogram plotted from 0 packet to 6,000 packets at every 150 packets. Figure 2.19 shows the histogram with REIsx ( $\lambda = 4.0$ ) on the random network. Because there is no nodes with more than 1,000 packets on REIsx ( $\lambda = 4.0$ ), the histogram plotted from 0 packet to 6,000 packets at every 150 packets is not shown. Every node sends 5 packets per ss for the random destination nodes. The number of packets from node 38 to node 45 is 32 packets per ss. The average cost of link (latency) is 4.0 ss. The packet processing limit is 100 packets/ss. When an average packet size is 400 bytes [48] [49], the ability of this packet processing limit is equal to 100k packets/ss, 320 Mbps of throughput. In the traffic condition, as the average link cost is 4 ss, and the packet processing limit is 100 packets/ss, we have no latency divergence if the total number of arriving packets is less than 400 packets per node. The average latency time increase constantly in OSPF, OSPF+ and OSPF++, but the average latency time in REIsx ( $\lambda = 4.0$ ) does not increase constantly.

In OSPF, there are not a few nodes holding more than 1,000 packets in their queues, while others hold much fewer than 100 packets. Nodes with too many packets will cause congestion. The most congested node holds over 296,000 packets at 6000 ss (Figure 2.14). Other nodes holds less than 425 packets. The number of the node that holds packets from 25 to 50 is the most.

In OSPF+, the link state is updated and change the routing table because of the increment of the latency at near 2400 ss and 4800 ss. The distribution of the number of the

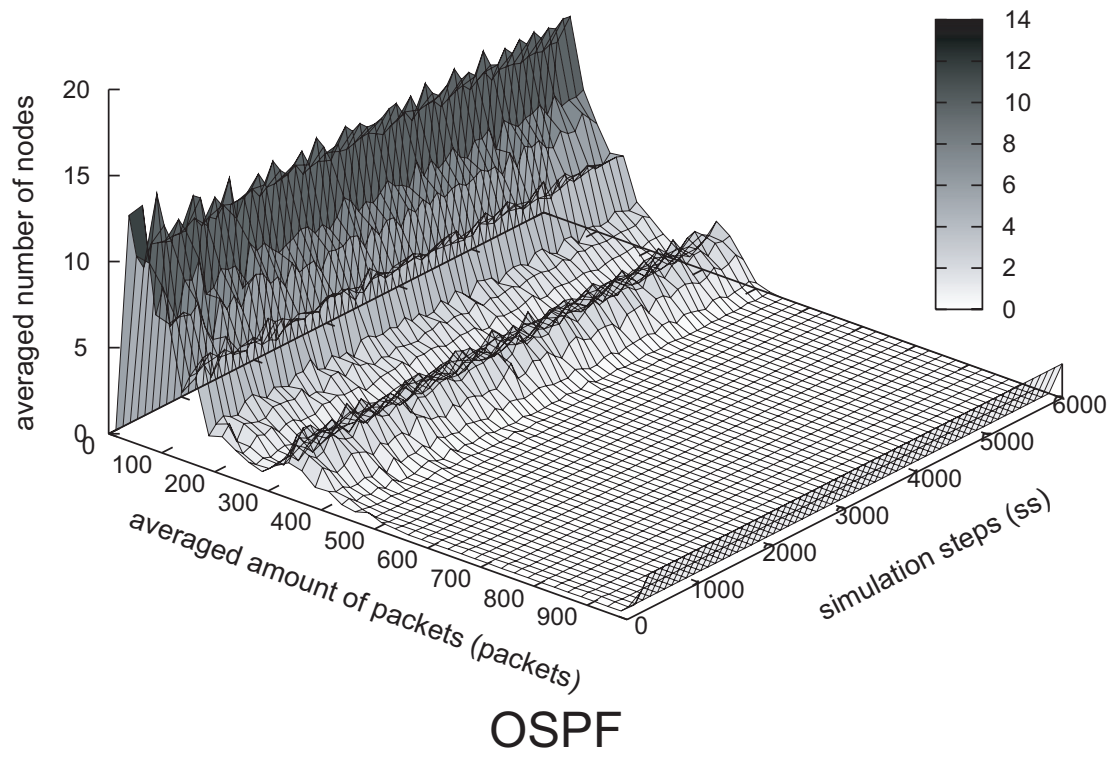


FIGURE 2.13: Distribution of queued packets with OSPF in Random Network.

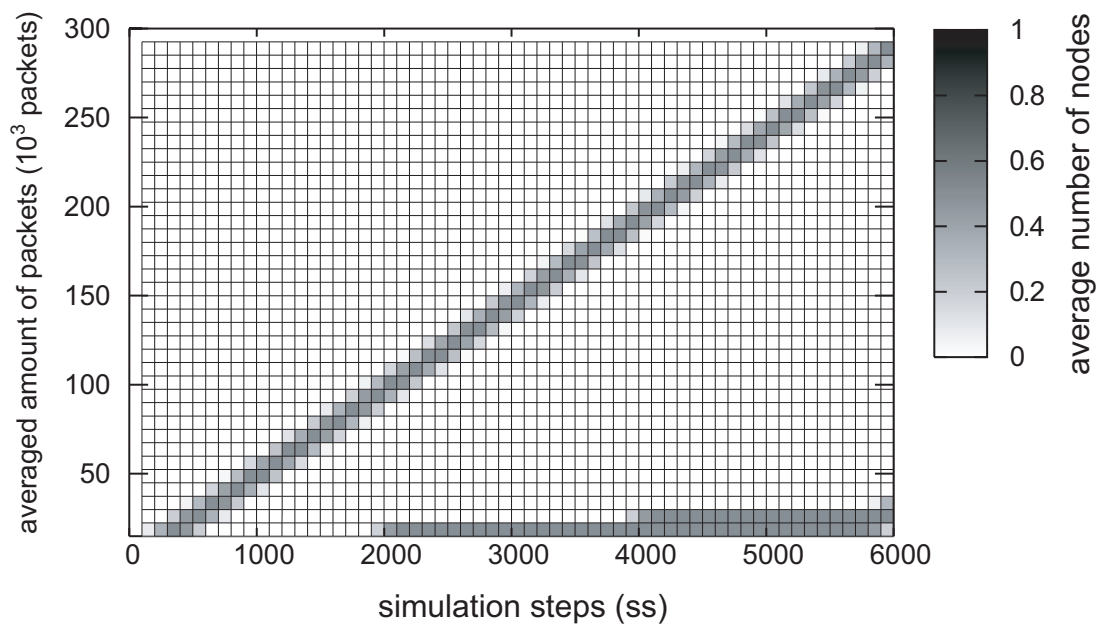


FIGURE 2.14: Distribution of queued packets in nodes. (OSPF)

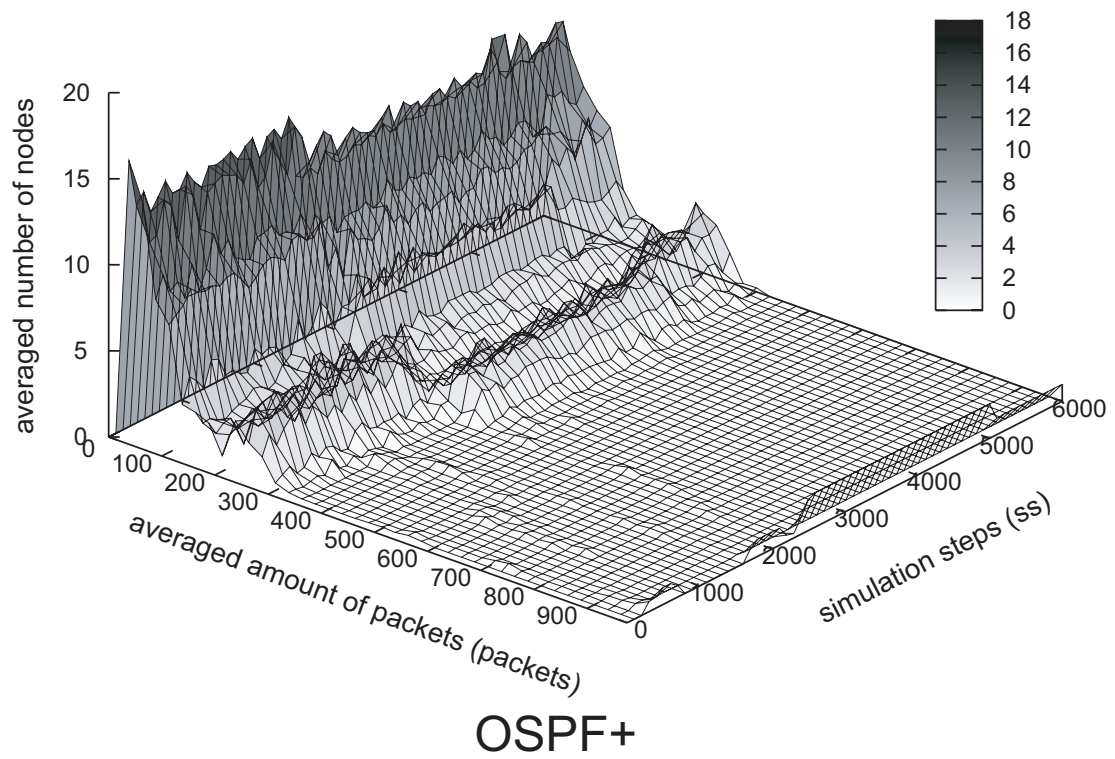


FIGURE 2.15: Distribution of queued packets with OSPF+ in Random Network.

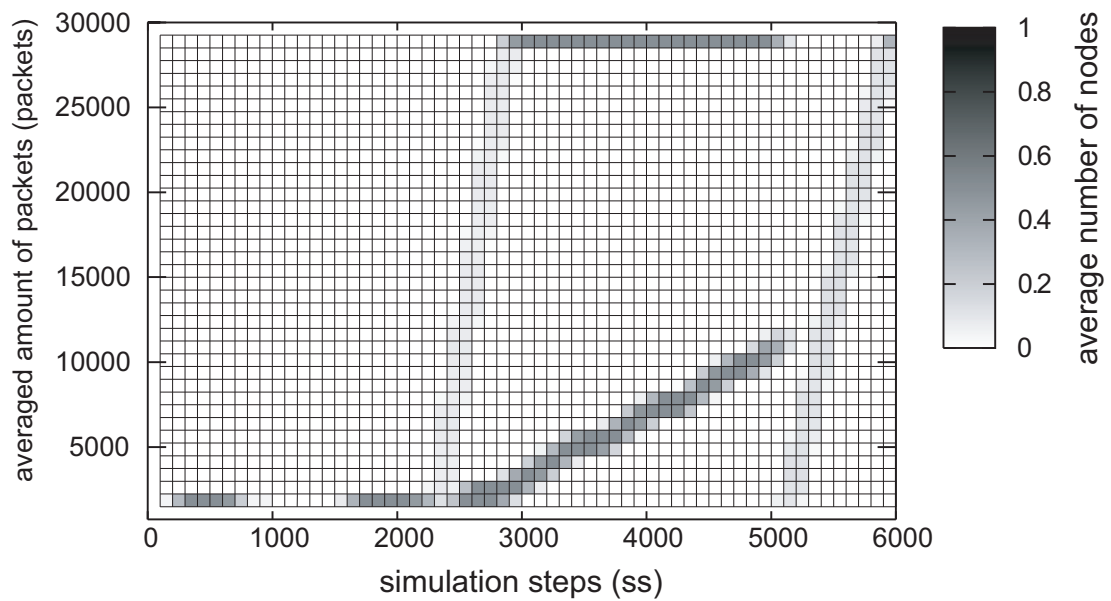


FIGURE 2.16: Distribution of queued packets in nodes. (OSPF+)

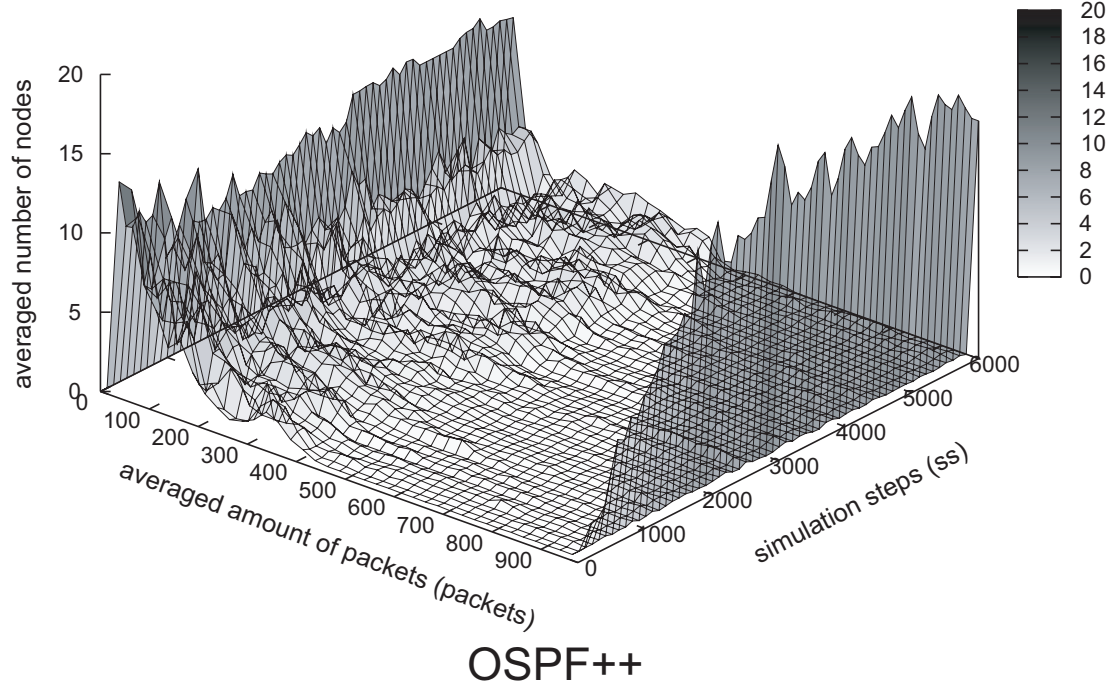


FIGURE 2.17: Distribution of queued packets with OSPF++ in Random Network.

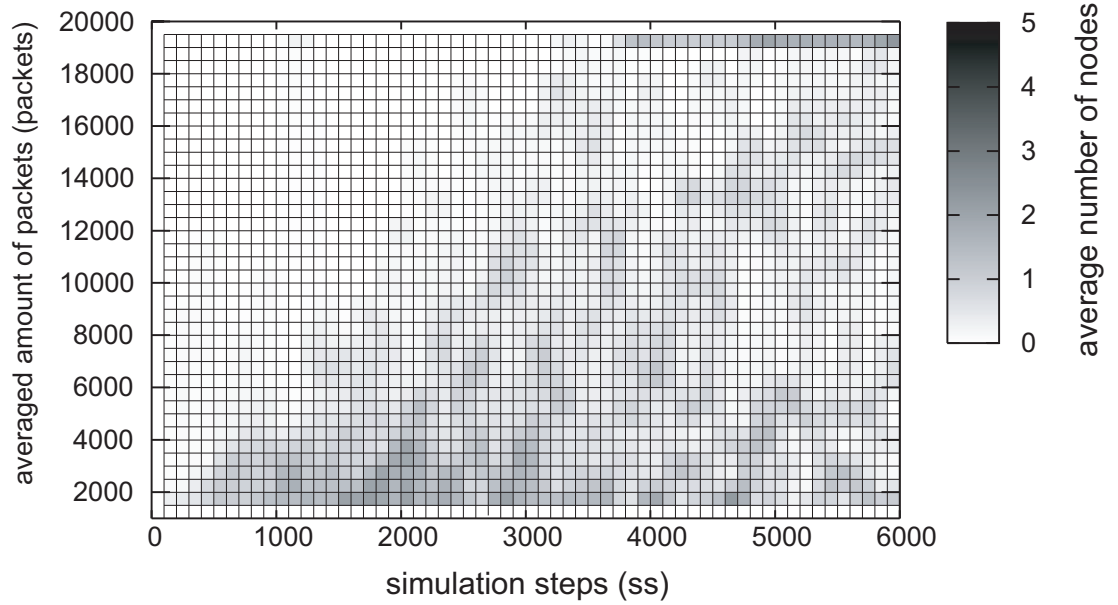


FIGURE 2.18: Distribution of queued packets in nodes. (OSPF++)

packets queued is shown to be changed (Figure 2.16). On the situation of the evaluation, after the change of the routing table, before the congested nodes is releaved and turn to be vacant, the routing table is changed again because of the increment of the latency on the other nodes. By the result, the problem of increment of the latency is not solved in the network because particular several nodes are congested in order.

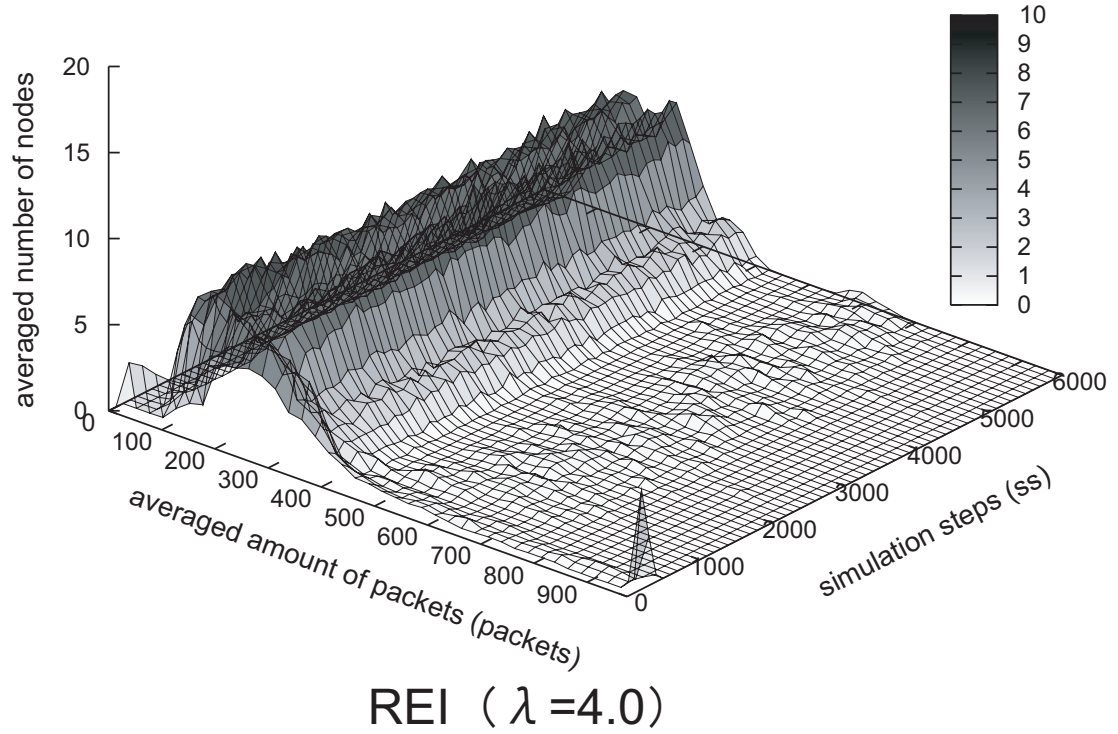


FIGURE 2.19: Distribution of queued packets with REIsx ( $\lambda = 4.0$ ) in Random Network.

On the result of OSPF++ is quite different from the result of OSPF and OSPF+ (Figure 2.18). The distribution of the number of the packets queued is changed very frequently. OSPF++ is seemed to avoid the large congestion in a few nodes by altering the routing table whenever any nodes has small scale congestion.

On the other hand, the distribution of queued packets in REIsx ( $\lambda = 4.0$ ) lies within the range of 100 to 400 packets. 87 % of the all nodes has 150 to 250 packets. We can say that REIsx realizes a good distribution of packets autonomously using probabilistic selection weighted by the observed latency. OSPF and

Figure 2.20 and Figure 2.21 illustrate histograms showing the amount of queued packets for routing at every node in OSPF++ and REIsx ( $\lambda = 4.0$ ) on the scale free network. In REIsx, although nodes that have over 1,000 packets exist from the start of the simulation up to 2000 ss, after 2000 ss, there are only nodes that have fewer than 1,000 packets. This shows temporal latency gain, as shown in Figure 2.11. Meanwhile, in OSPF++, 52 % nodes hold more than 1000 packets and 60 % of other nodes hold less than 75 packets. OSPF++ cannot realize effective load balance.

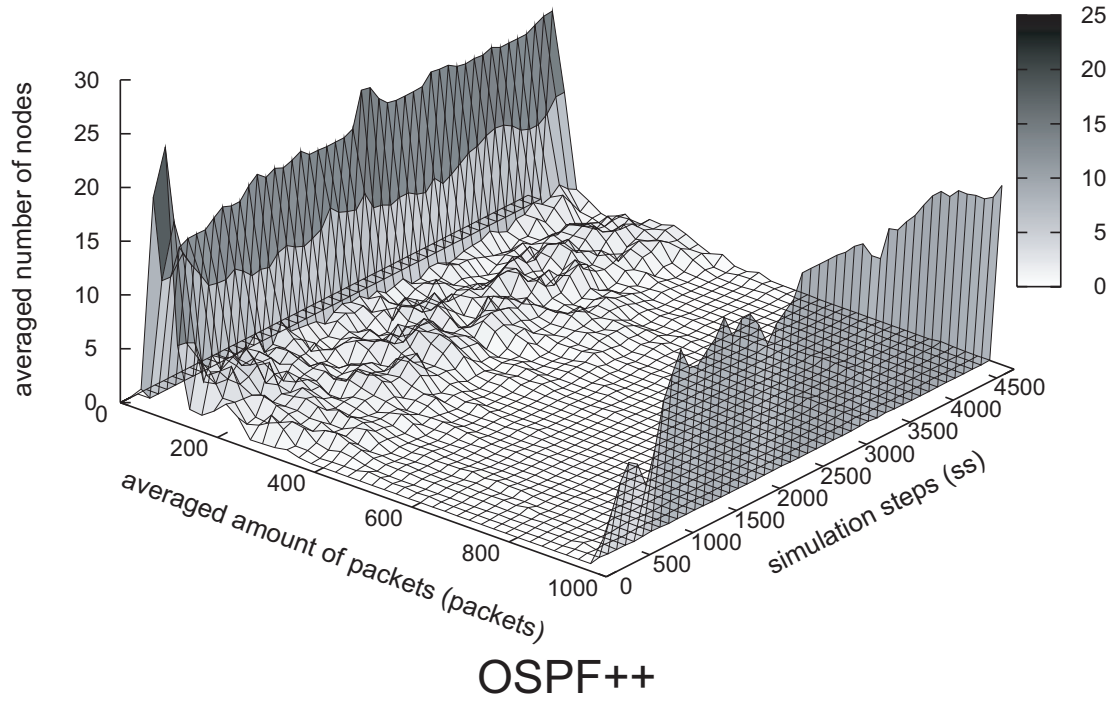
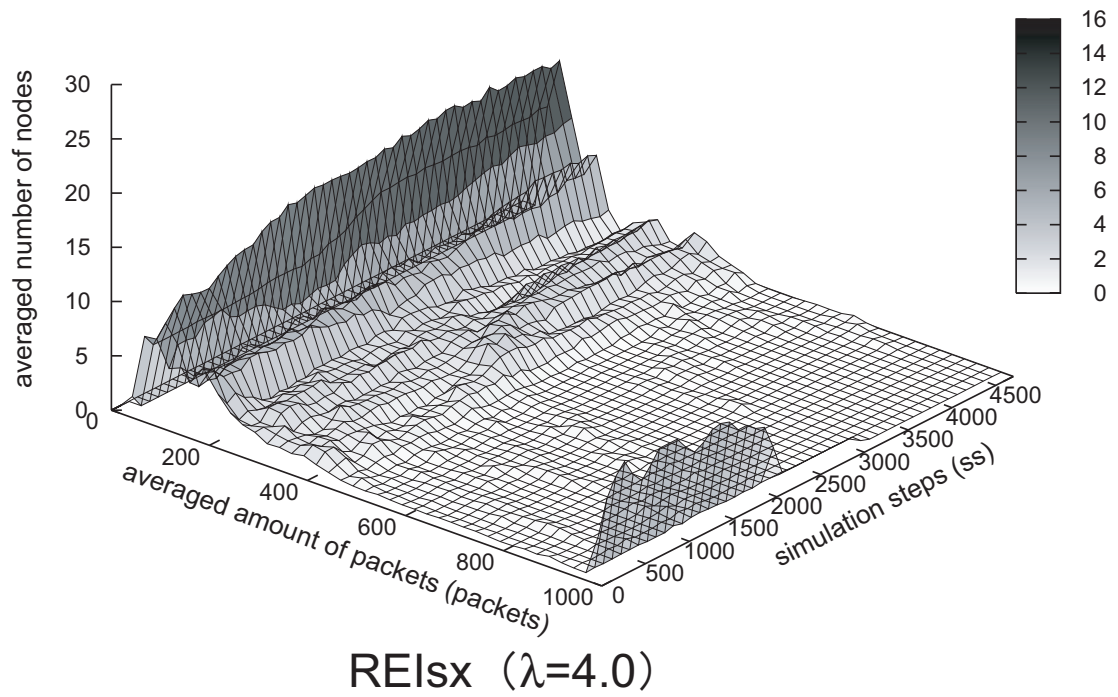


FIGURE 2.20: Distribution of queued packets with OSPF++ in SFN.

FIGURE 2.21: Distribution of queued packets with REIsx ( $\lambda = 4$ ) in SFN.

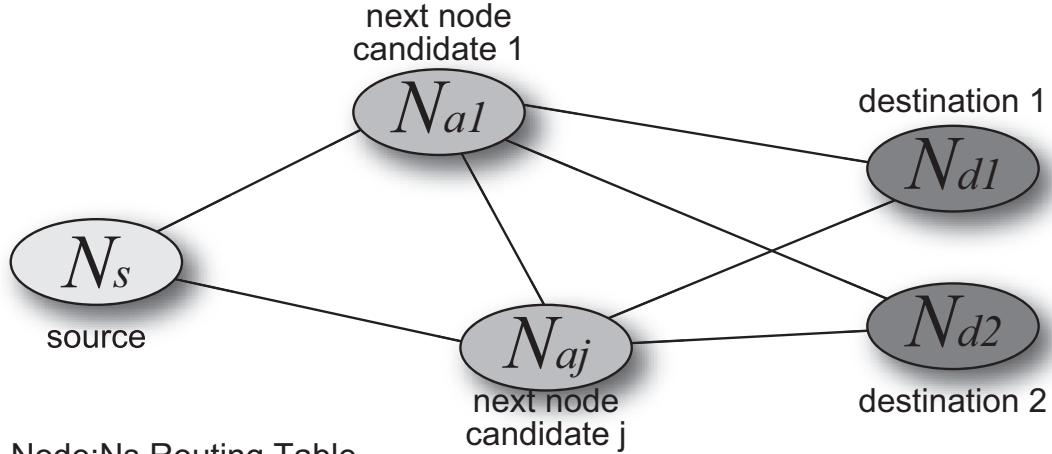
## 2.4 Experimental Evaluation in IP network

Though the round-trip time (RTT) is generally known as a standard for a network delay time, it lacks a sufficient explanation for use as an evaluation value for network routing because it performs the delay time when a packet makes a round trip, as the name suggests, and not a one-way trip. Using methods for a dramatic improvement in the time synchronization through Network Time Protocol (NTP) accuracy [50, 51], we can easily observe a one-way delay between one node and an adjacent node. In addition, we can also presume network conditions such as congestion and available bandwidth using a one-way delay. Although measurement packets and other traffic should be separated and independent in the network to achieve a high-quality one-way delay measurement, NTP can synchronize in 100 micro seconds order on high utilization of network by using periodic broadcast packets of NTP [52]. In a broad area network, a one-way delay obtained using NTP synchronization is sufficiently accurate for evaluating the paths. In this paper, we therefore use a one-way delay with NTP synchronization and evaluate a TE method for an adaptive traffic distribution. NTP packets are also used to measure a one-way delay. In this method, a routing table is constructed from this delay information.

### 2.4.1 A routing table and a probe packet

Each node has a unique routing table for all destination nodes based on its adjacent nodes. In addition, any adjacent nodes,  $N_a$ , (equal to all next node candidates) for any destination nodes,  $N_d$ , have a score sequence with a length of  $w$ . This score is expressed as  $L_{N_d N_a} = \langle S_{N_d N_a}^1, S_{N_d N_a}^2, \dots, S_{N_d N_a}^w \rangle$ . This score,  $S_{N_d N_a}^j$ , shows a one-way delay in which a node is obtained from a measurement packet where the node is sent to any destination node,  $N_d$ , through an adjacent node,  $N_a$  (Fig. 2.22).

Each node sends measurement packets to any destination node  $N_{dst}$  periodically through all adjacent nodes,  $N_{a_1} \dots N_{a_j}$  (where  $j$  is the total number of adjacent nodes), and measures the one-way delay from itself through the next node candidates to  $N_{dst}$ . In the measurement packet header, the transmission time,  $t_{send}$ , is described. The time of arrival,  $t_{recv}$ , is also described when the packets reach their destination. The packets with both the transmission and arrival times described in its header are sent back to the source node, and the difference value between the time of transmission and arrival,  $d = t_{recv} - t_{send}$ , is added to the head of the score sequence corresponding to the packet destination node,  $N_{dst}$ , and adjacent node  $N_{a_n}$ , that is passed through. Before the new score is added into the score sequence, the index number of all elements in the score sequence are added to one score, and a score that exceeds the length of  $w$  of the score

Node:  $N_s$  Routing Table

<i>destination</i>	<i>next node</i>	<i>scores</i>
$N_{d1}$	$N_{a1}$	$S_{N_{d1}N_{a1}}^1, S_{N_{d1}N_{a1}}^2, \dots, S_{N_{d1}N_{a1}}^{w-1}, S_{N_{d1}N_{a1}}^w$
	$N_{aj}$	$S_{N_{d1}N_{aj}}^1, S_{N_{d1}N_{aj}}^2, \dots, S_{N_{d1}N_{aj}}^{w-1}, S_{N_{d1}N_{aj}}^w$
$N_{d2}$	$N_{a1}$	$S_{N_{d2}N_{a1}}^1, S_{N_{d2}N_{a1}}^2, \dots, S_{N_{d2}N_{a1}}^{w-1}, S_{N_{d2}N_{a1}}^w$
	$N_{aj}$	$S_{N_{d2}N_{aj}}^1, S_{N_{d2}N_{aj}}^2, \dots, S_{N_{d2}N_{aj}}^{w-1}, S_{N_{d2}N_{aj}}^w$
$N_{a1}$	$N_{a1}$	$S_{N_{a1}N_{a1}}^1, S_{N_{a1}N_{a1}}^2, \dots, S_{N_{a1}N_{a1}}^{w-1}, S_{N_{a1}N_{a1}}^w$

FIGURE 2.22: Routing table with scores.

sequence is then broken off. That is, the newest  $w$  scores are constantly held in the score sequence constantly (Fig. 2.23).

In the measurement packets, the sequential number is also described. A node that transmits measurement packets can identify any packets by their sequential number. If  $d_{limit}$  milliseconds pass since a measurement packet has been transmitted, the packet is considered a lost packet. A penalty delay time,  $d_p$ , is then added to the head of the score sequence.

After measurement packets are sent to adjacent nodes, the packets are brought to the next node according to a selection rule hop by hop, and finally arrive at the destination node. To raise the precision of a one-way delay for a neighborhood node, nodes measure the delay using a source routing option in an IP and setting an explicit path for nodes to which the number of hops is less than or equal to  $h_n$  from own node. In this case, this explicit path is a concatenation path  $[N_s, N_{a_n}, \dots, N_{dst}]$ , and is constructed from the source node,  $N_s$ , and a path  $[N_{a_n}, \dots, N_{dst}]$  that has a minimal number of hops from an adjacent node,  $N_{a_n}$ , to the destination node,  $N_{dst}$ . If there are multiple paths for one

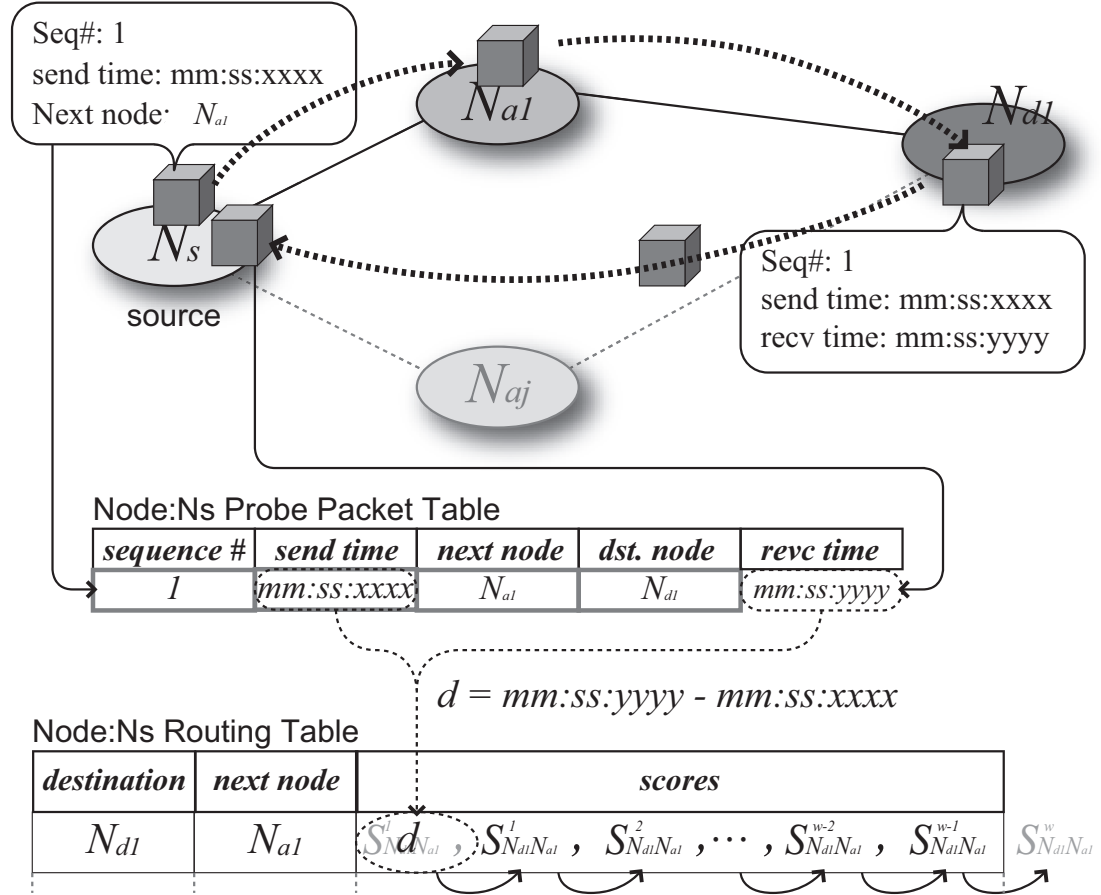


FIGURE 2.23: Protocol for probe packets and updating routing table.

adjacent node,  $N_{a_i}$ , a one-way delay measured from each path is also pushed into the score sequence corresponding to the adjacent node,  $N_{a_i}$ , and destination node,  $N_{dst}$ .

#### 2.4.2 Algorithm for the next node selection

When a packet reaches a node, if the node is not the destination node of the packet, the node chooses the next node for the packet by referring to the score sequence for the destination node in its own routing table. Next node candidates are all adjacent nodes,  $N_{a_i}$  ( $1 \leq i \leq r$ ,  $r$  is the total number of adjacent nodes).

If there are more than two candidates, a representative value,  $Q_{a_i}$ , of a score sequence for destination node,  $N_d$ , and candidate node,  $N_{a_i}$ , is given as  $Q_{a_i} = \sum_{m=1}^w C_m S_{N_d N_{a_i}}^m$ , which is the weighted average for the score sequence. In this equation,  $C_m$  is the linear weight function, which has a negative gradient.

$Q_{a_i}$  indicates the weighted average time of a one-way delay. Though the minimum  $Q_{a_i}$  in all candidates is desirable, it will be difficult to accept a larger amount of traffic in

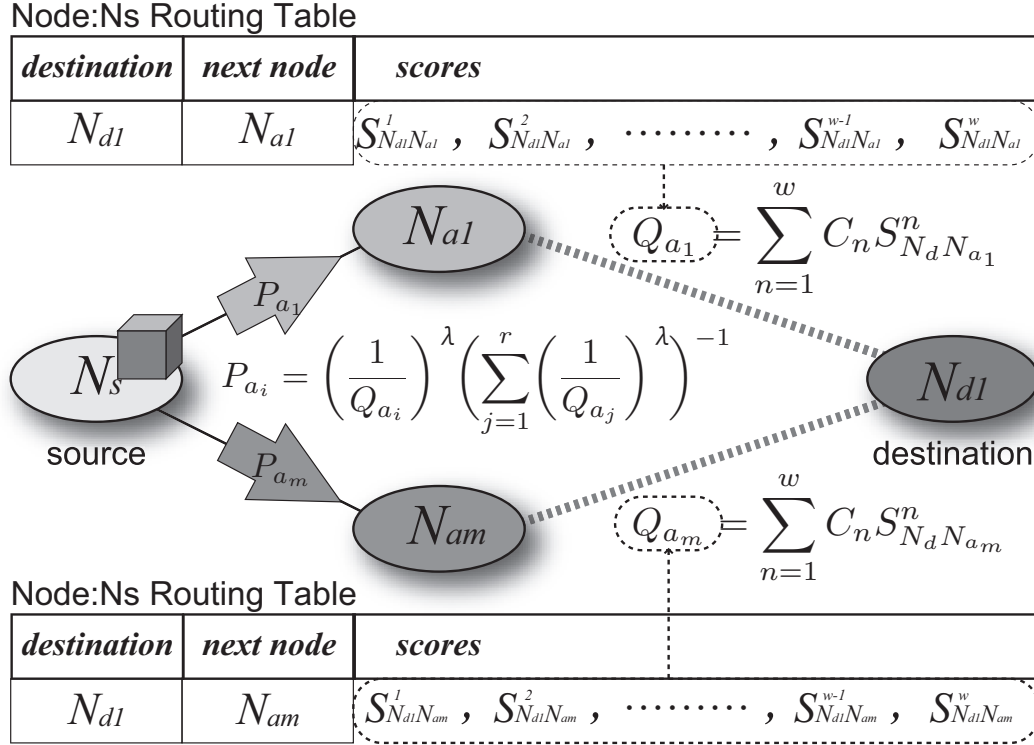


FIGURE 2.24: Probabilistic selection of the next-node.

a network through a deterministic next node selection that does not distribute network traffic widely. The next node is therefore chosen randomly, weighted by its representative value,  $Q_{a_i}$ , raised to the  $\lambda (> 0)$ -th power (Fig. 2.24). That is, probability  $P_{a_i}$ , in which a candidate node,  $N_{a_i}$ , is selected as a next node, is expressed through equation (2.4).

$$P_{a_i} = \left( \frac{1}{Q_{a_i}} \right)^\lambda \left( \sum_{j=1}^r \left( \frac{1}{Q_{a_j}} \right)^\lambda \right)^{-1} \quad (2.4)$$

Parameter  $\lambda$  which controls the selection probability for the next node, raises the probability of a candidate with a relatively lower representative value. When  $\lambda \rightarrow \infty$ , the candidate node with the most lowest representative value is expressly selected as a next node. In this paper, we describe the method to determine a next node using the parameter  $\lambda$  and equation (2.4), i.e., Network adaptive Routing algorithm for Environmental Intelligence (NREI).

Because a measurement packet for a coupling of source node,  $N_s$ , and destination node,  $N_{dst}$ , in which the number of hops is more than  $h_n$  also measures a one-way delay, various values of a one-way delay with various routes are added into the score sequence

for any adjacent nodes,  $N_{a_n}$ , and destination node,  $N_{dst}$ . As a result, if the distribution of one-way delay around the whole network is not equivalent, and if there is an imbalance of one-way delay in the network, the imbalance seems to be smoothed from nodes far from the imbalance lines by the weighted average of the scores. Conversely, the proposed algorithm tends to avoid sections of paths with higher delay.

### 2.4.3 Experimental settings

To evaluate the effectiveness of the proposed method, we created adaptive routers (AR) implementing the proposed algorithm, and conducted two types of evaluation experiments on the IP network testbed.

An AR is implemented on a PC with a 1.86 GHz Xeon CPU, 512 MB of memory, and a NetBSD 4.0.1 operating system. An AR measures the one-way delay, creates the routing tables, and transmits packets using the distribution ratio given by the routing tables. To measure the one-way delay, an AR sends and receives UDP packets that are described in the time stamp of its payload. In an AR, there are three types of process, as listed below.

1. sending measurement packets.
2. receiving measurement packets.
3. making routing table.

Processes (1) and (3) are run in the measurement of AR1, and process (2) is run in the measurement of AR2. Process (1) has features for sending UDP packets whose time stamp is described through the path assigned by the AR in the source node. To raise the availability of the score sequences, the source node assigns the measurement packets whose destination node is within less than  $h_n$  hops using the Strict Source Routing and Record (SSRR) option of an IP. Process (2) has features for receiving measurement packets, adding a time stamp, and sending the packets back to the source node. Process (3) calculates the one-way delay of the measurement packets, constructs the routing table, transmits information regarding the distribution ratio calculated from the routing table, and then updates the data plane.

The data plane is implemented in the OS kernel of an AR. This feature has its own routing table that differs from the standard routing function of its OS. In this table, lists of next hops with their distribution ratio are described for the IP prefix of all destinations. The data plane selects the longest match prefix for the destination address

of each inbound IP packet. Then, according to its distribution ratio, the next node is selected and weighted randomly. If there is no prefix matched with an inbound packet in the AR routing table, the next node is selected using the standard routing function of its OS.

The parameter  $\lambda$  in equation (2.4), which affects the distribution ratio, raises the selection probability of relatively better nodes among the next node candidates. When  $\lambda \rightarrow \infty$ , the best next node candidate is selected at a rate of 100 %. That is,  $\lambda$  can change the degrees between deterministic selection and random selection. In this implementation, an operator can change parameter  $\lambda$  as needed. To distribute the traffic demands according to equation (2.4) accurately, a per-packet routing method is adopted in the implementation.

Too high a frequency of a one-way delay measurement results in too much accumulation of microscopic delay information. Based on the experimental evaluations, a large change in a one-way delay on the order of 10 ms can be considered a rare case. We set the measurement frequency to every 100 ms. It is difficult to measure the change in total delay during frequent calculations of the distribution ratio every 1 s. On the other hand, at a lower frequency, this method cannot sufficiently respond to environmental changes in the network. We therefore set the frequency to every 10 s. During a period of at least 10 s, the traffic demands are distributed by the most recent calculation. To reduce frequent changes in the distribution ratio, the size of score sequence  $w$  is set to 1,000, and thus the next calculation will be affected by past changes in the distribution ratio. In addition, the penalty value of delay  $d_p$  was set to 1,000 ms in all evaluation experiments so that the representative values did not change drastically by only a single penalty (packet loss) in the distribution ratio calculation. To give priority to the processing ability of the data plane, the distribution ratio is rounded to the nearest 10 % so that the processing ability does not run short when the frequency of the distribution ratio calculation is raised.

For the evaluation experiments, we set up ARs in four research institutes.

- AR1: Sapporo (Hokkaido University: 43° 07' N, 141° 34' E)
- AR2: Toyama (Intec Inc.: 36° 70' N, 137° 21' E)
- AR3: Yamanashi (Yamanashi Prefectural University: 35° 66' N, 138° 55' E)
- AR4: Kochi (Kochi University of Technology: 33° 62' N, 133° 72' E)

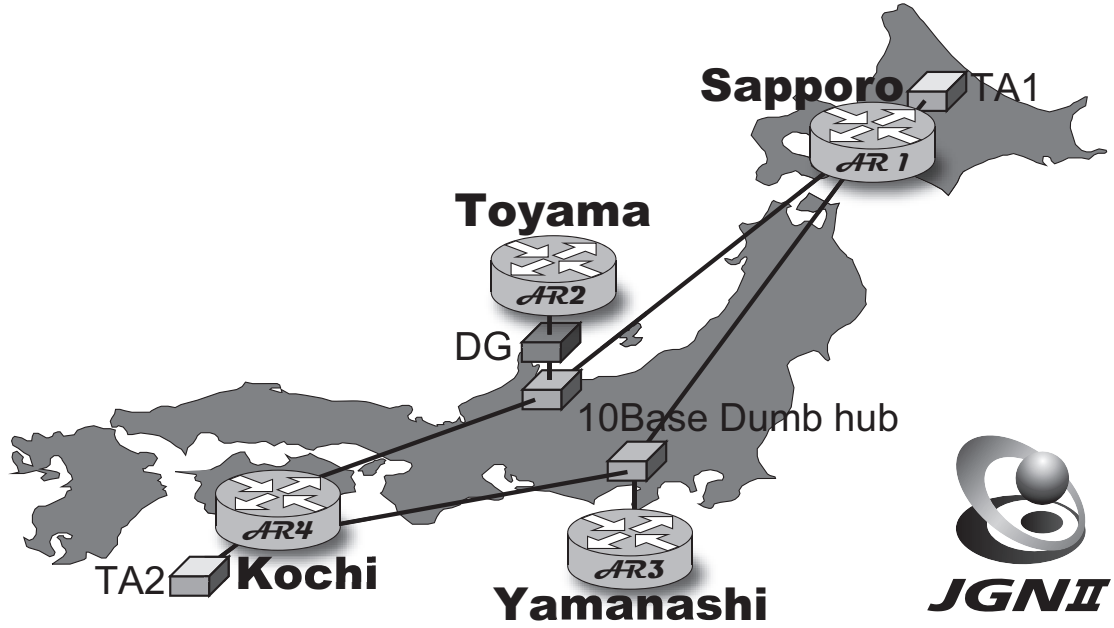


FIGURE 2.25: Network construction for evaluation experiments.

The research institutes were connected to each other using Japan Gigabit Network eXtreme: New Generation Network Testbed (JGN-X)<sup>1</sup> and Regional Internet Backbone (RIBB) [53]. The ARs in each institutes were connected through IPv4 over IPv4 tunneling to make an overlay network with the ARs. In the overlay network and its topology, each AR can use the other ARs for the next node. In this evaluation experiment, we created user networks in Sapporo (AR1) and Kochi (AR4), and set traffic agents TA1 and TA2 in each user network. Traffic between the two user networks was routed to either the paths going through Toyama (AR2) or that going through Yamanashi (AR3). That is, the traffic was distributed into the two paths shown below.

1. AR1 (Sapporo) - AR2 (Toyama) - AR4 (Kochi)
2. AR1 (Sapporo) - AR3 (Yamanashi) - AR4 (Kochi)

Traffic agents measure the total delay time based on a distinction of each path. AR1 and AR4 decide next node to AR2 or AR3 according to the distribution ratio calculated using the one-way delay information based on the measurements. In AR2 and AR3, congestion is produced purposely by constraining the network bandwidth to observe the traffic control. AR2 and AR3 are connected to JGN-X with 10 Mbps half-duplex lines. In addition, a delay generator (DG) was set between AR2 and the JGN-X line, which can control the delay on the order of milliseconds (Figure 2.25).

<sup>1</sup><http://www.jgn.nict.go.jp>

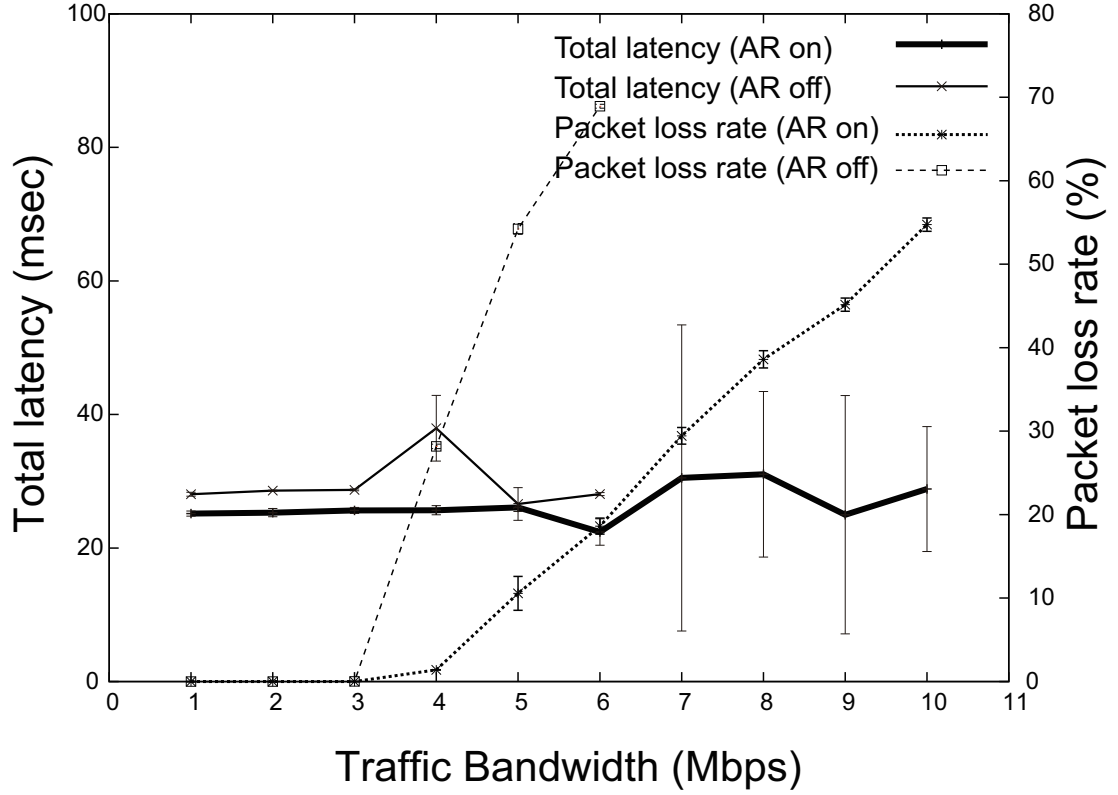


FIGURE 2.26: Changes of delay and packet loss ratio for increment of network traffic

#### 2.4.4 Experiment 1. Raising the total amount of traffic demands

To confirm the ability of load distribution in an overlay network consisting of the proposed ARs, we examine a situation in which the amount of traffic demands exceeds the bandwidth of a single path. To set up this situation, TA2 connected to AR4 generates UDP traffic demands to TA1 connected to AR1. We measured the changes in the one-way delay in each line and the loss rate of packets when changing the amount of traffic demand from TA2 to TA1. We then compared the results with AR features enabled to those with AR features disabled, in which all traffic demand goes through only a single path. To generate the traffic demands and measure the loss, *iperf*<sup>2</sup>, which can be used to analyze the network bandwidth and throughput, is utilized. By changing the size of the transmitted packet and the transmission interval, the amount of traffic demand is controlled. In generating the traffic demand from TA2 to TA1, a one-way delay is measured at TA1 and TA2 using a synchronized clock by NTP. We set parameter  $\lambda$  to 4.0, which can determine the property of the distribution ratio toward the next node, and change the amount of traffic demand from 1 to 10 Mbps in 1 Mbps steps. A one-way delay and packet loss ratio were measured ten times in each step. Parameter  $h_n$ , which

<sup>2</sup><http://dast.nlanr.net/Projects/Iperf/>

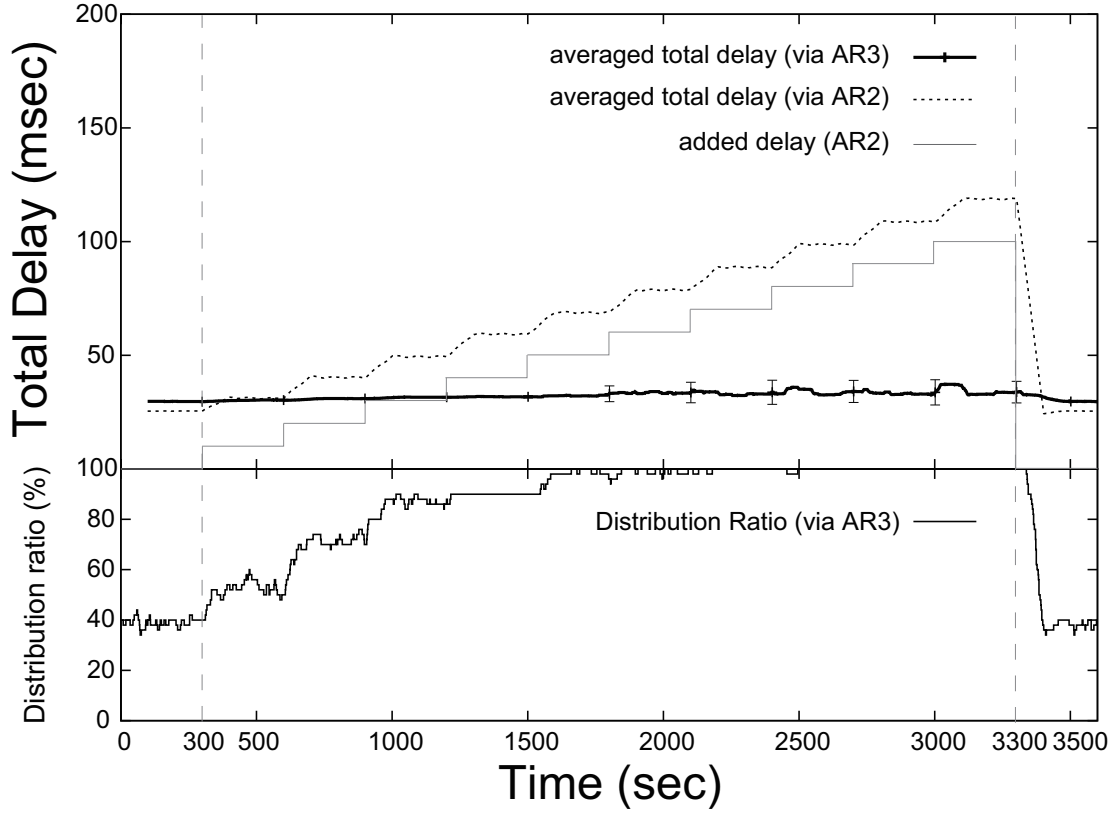


FIGURE 2.27: Changes of packet distribution for path delay increment (3 Mbps)

defines the scope using SSRR to measure the one-way delay, is set to 2.0. Figure 2.26 shows the delay and packet loss ratio for an increment of traffic demand.

Packet losses were measured by disabling the AR initially at 3 Mbps because AR2 and AR3 are connected through a 10 Mbps repeater switch. Iperf cannot be used to measure the packet loss ratio at 7 Mbps. An increase in the total delay and jitter was measured at 4 Mbps. On the other hand, when enabling the AR, the packet loss ratio was reduced to 5 % of the ratio when disabling the AR at 4 Mbps. Iperf also cannot be used to measure the packet loss ratio at 11 Mbps. An increase in jitter was measured at 7 Mbps, but the total delay was not changed considerably. By enabling the AR, traffic is distributed properly. A redundant network stream with forward error correction (FEC) using a Reed-Solomon code can be used to recover the packet loss ratio to 2.5 % from a ratio of 20 % [54]. With that, when we consider this 20 % packet loss ratio as an index of a healthy network condition, the proposed algorithm can tolerate double the amount of traffic in this experimental network by distributing the traffic autonomously.

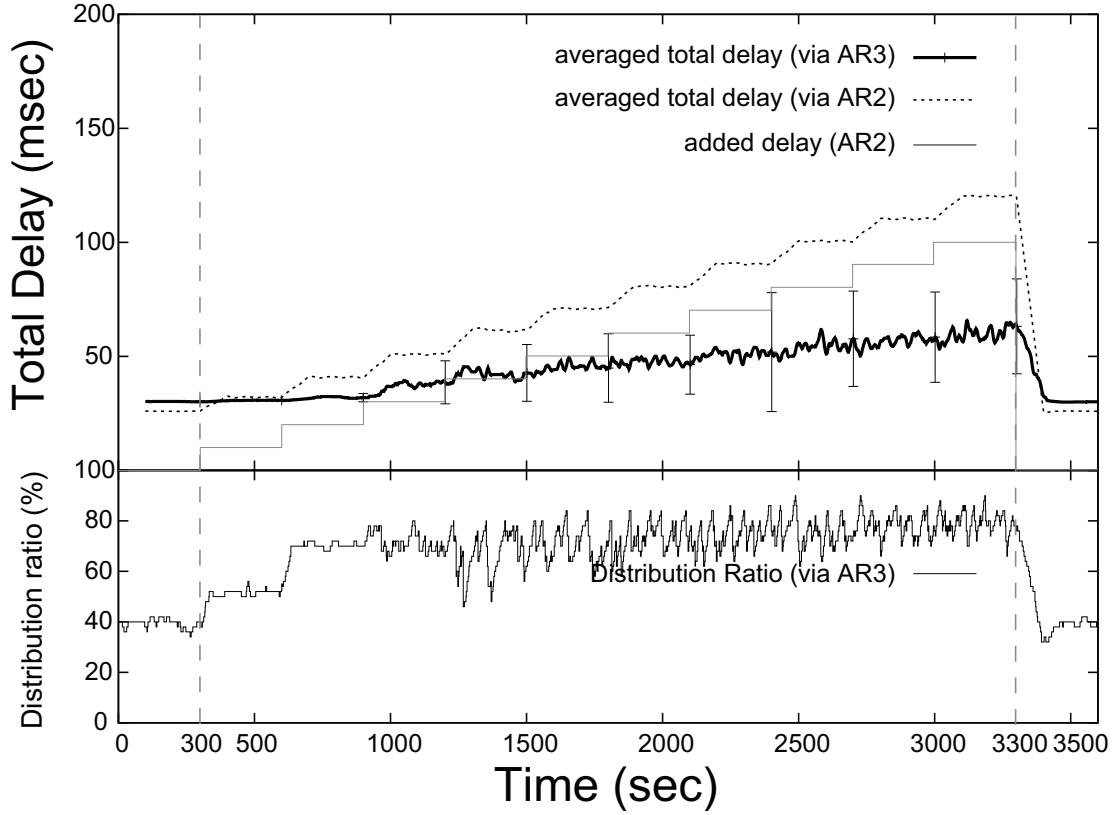


FIGURE 2.28: Changes of packet distribution for path delay increment (4 Mbps)

### 2.4.5 Experiment 2. Raising the delay on one side

For events such as a regional HD movie transmission, which generates a huge amount of traffic, the quality of the network repeaters or network switches frequently worsens because the network instruments at different institutions are not always heterogeneous. During such failures, it is recommended to change the distribution ratio toward the next node autonomously without a disconnection, and not arrange the ratio manually. To evaluate the autonomy of the ARs, we examined a situation in which the AR can change the ratio against an increment in the path delay.

The total delay of path AR1 - AR2 - AR3 is increased purposely from 0 to 100 ms in steps of 10 ms using the delay generator of AR2, and the added delay is then returned to 0 ms. During the change, TA2 connected to AR4 generates UDP traffic demand to TA1 connected to AR1. We measure the changes in the ratio of distribution on AR4 and the changes in the total delay reaching AR1 after going through both AR2 and AR3. The traffic amounts were 3 Mbps, 4 Mbps, and 5 Mbps and we set parameter  $\lambda$  to 4.0. The evaluation experiments were performed five times for each traffic amount. Figure 2.27 shows the changes in total delay and the ratio of distribution. The changes in total

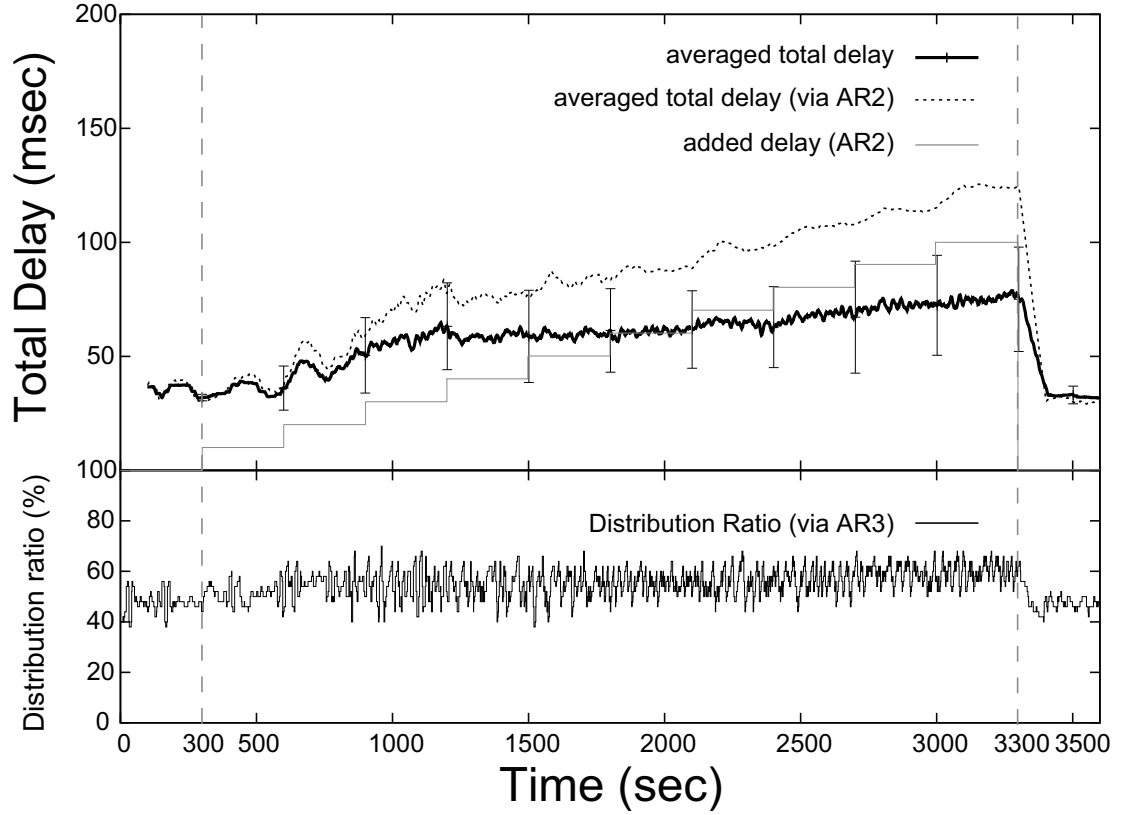


FIGURE 2.29: Changes of packet distribution for path delay increment (5 Mbps)

delay after going through the path through AR3 are shown with the standard variation in steps of 300 s.

Without a purposeful delay [0, 300 sec], the average total delay going through the path through AR2 is 25.5 ms, and that through AR3 is 29.7 ms. The theoretical ratio calculated using equation (2.4) based on these averages is  $\text{AR2:AR3} = 1.00:0.54$ . In this implementation, the actual ratio is  $\text{AR2:AR3} = 60:40\%$  after rounding to the nearest 10 %. This ratio fits the value measured. For a traffic amount of 3 Mbps, there is no network congestion when all traffic is concentrated in only one path (Figure 2.27). At 300 s, the total delay in the path going through AR2 is escalated. At the same time, the ratio of distribution in AR2 is also escalated when selecting AR3. At 1,500 s, when the delay generator creates an additional delay of 50 ms, the path through AR3 is selected at a rate of 100 %. The additional delay returns to 0 s in 3,300 s, and the ratio of distribution also returns to 60:40 %.

When the traffic amount is 4 Mbps, network congestion occurs and some packets are lost with a disabling of the AR features (Figure 2.28). When the ratio of distribution is  $\text{AR2:AR3} = 20:80\%$ , the amount of traffic going through the path through AR3 is more than 3.2 Mbps. The penalty latency is added to the score sequence because of the

packet loss. An oscillation in the ratio of distribution begins at 1,200 s when the delay generator creates an additional delay of 40 ms. The average ratio of distribution between [1,200, 3,300] is 73.9 % (2.92 Mbps), and the standard variation is 7.15 %. Until just before the packets are lost in the path through AR3, the ratio of distribution through AR3 is increased. In the results, the delay time through AR3 is also loosely increased because the transmission ability is exceeded. At 2,400 s, the maximum standard value is 26 ms, and the path through AR2 is selected at a rate of about 50 %.

By the same token, when the traffic amount is 5 Mbps, network congestion occurs, and some packets are lost with a disabling of the AR features (Figure 2.29). The average ratio of distribution between [1200, 3300] is 56.5 % (2.82 Mbps), and the standard variation is 5.32 %. Until just before the packets are lost in the path through AR3, the ratio of distribution through AR3 is increased and converges. Because the penalty latency is bigger than the latency added by the delay generator, a packet loss is avoided and thus the ratio of distribution is determined.

## Chapter 3

# Centralized deterministic approach

### 3.1 Objective

Traditional TE methods can be classified into two types: offline TE methods and online TE methods. In offline TE methods, traffic demands are distributed according to the maximum size of the traffic demands estimated from the measured network information. In online TE methods, the demands are distributed on a real-time basis according to the network information observed [17]. In delivering a movie, the available bandwidth of the communication lines that connect the facilities to each other should be estimated because the quality of the movie is profoundly affected by packet losses. In this research, we adopt an online TE method.

MPLS-TE, which is an online TE methods is used in high production levels such as a backbone network within a network carrier [19]. Some problems are incurred for interoperability in MPLS router implementations [23]. It is difficult for various inhomogeneous organizations to adopt this TE approach to connect organizations and facilities to each other equally. OSPF and BGP parameter settings have also been proposed [55, 56]. Although this approach in changing the routing parameters according to the network information is measured in real-time, it is difficult for this approach to utilize a multi-home environment. When using a traditional static routing algorithm, it is difficult to distribute the traffic demands flexibly.

To distribute the traffic demands in a small granularity, per packet routing is useful [24]. It is necessary for this type of routing to focus on reducing the packet reordering and avoid loop routing [25]. To reduce the packets reordering, it is useful for end hosts to

prepare a large buffer and correct lost packets with redundant data, such as in Forward Error Correction (FEC) [54]. In the case of an increasing number of facilities and data amounts to be delivered, the number of calculations of FEC for encoding and decoding increases greatly. The length of the delay time among the facilities is a serious problem, especially duaring real-time interaction and operation.

In a multi-home environment, it is necessary for TE methods with lower layer routing and switching to share the network operations among different organizations and facilities. These methods may also need network routers and network switches implemented using specified protocols. In addition, the cost problem is very serious during operation. In a TE method with overlay routing, a virtual network can be constructed using popular traditional hardware without increasing the operating costs, and without being constrained by the topology of the lower layer (physical) network. In overlay network routing, problems regarding network quality and stability occur because an overlay network is affected based on the lower layer network. It is highly necessary for TE methods that use an overlay network to measure the network information, such as a link down of the lower layer network, and the packet loss rate of the virtual paths. We also proposed the use of per-packet adaptive routing with using network latency information [57]. We then implemented this algorithm on PC routers, constructed an overlay network using these routers in a Regional Internet Backbone (RIBB) [53] network on the Japan Gigabit Network (JGN-X), and evaluated the superiority of this proposed algorithm by delivering movie data over a real network [58].

In accordance with the research backgrounds described above, we propose a TE method for delivering movie data among approximately ten organizations and facilities using an overlay network to realize flexible, real-time oriented online TE. This proposed method can follow changes in a lower layer network environment adaptively by measuring the lower layer network information and determining a combination of paths based on this information. Through a simulation evaluation, we show that this method selects combinations of paths that can reduce the packet loss ratio and total latency time of the packets. In addition, we show that this methods can derive such combinations within a realistic limited timeframe without difficulty of network operation using large computing resources of a cloud computing environment.

## 3.2 Basic Framework

The proposed TE method is intended for implementation on an overlay network consisting of routers of several facilities connected to adjacent routers with more than one independent communication line. Initially, we constructed a logical network topology

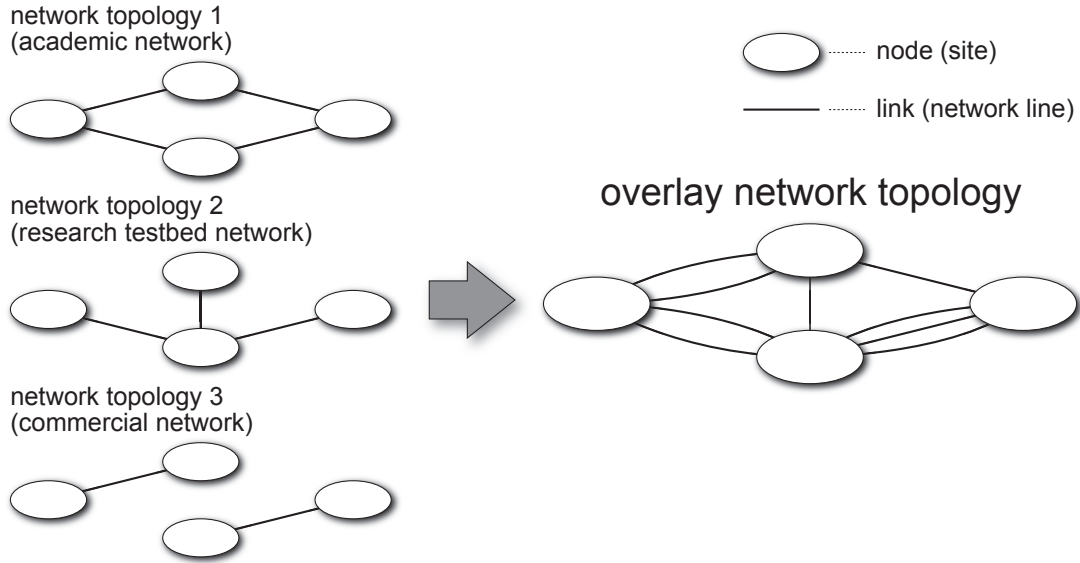


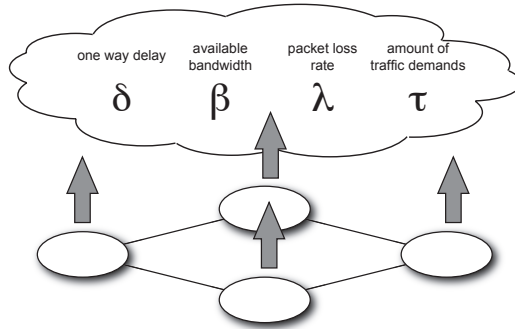
FIGURE 3.1: Overlay Network Topology.

of an overlay network. In academic computer networks and research network testbeds where the network topology is public knowledge, we use this topology as a part of the logical topology. In networks in which the topology is not public knowledge, we use a method of topology estimation through end-to-end latency measurements [59]. A logical topology is constructed by overlapping this topology information. We show this topology as a multi-directed graph in Fig. 3.1.

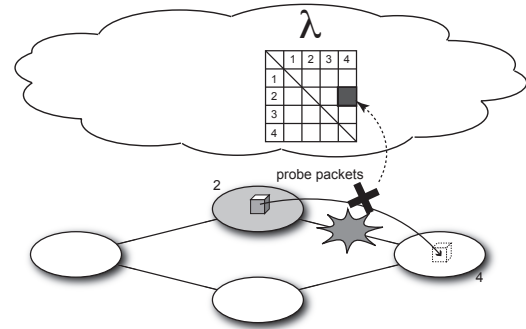
To control the adaptive routing (AR) on overlay network, we connect an AR router to the gateway of each facility. An AR router can route network packets towards any adjacent AR routers and can measure the one way delay, packet-loss rate, and available bandwidth between itself and another adjacent AR router. An AR router can also measure the amount of traffic demand toward any other routers, and only routes traffic demands among this overlay network. Other traffic demands are routed or switched according to the lower layer protocols.

We use a control service to evaluate combinations of paths and advertise the updated routing table derived from this calculation. This service is implemented on a cloud computing environment on the Internet because the service requires a large amount of computing power and should be separated from an overlay network to avoid a single point of failure (SPoF). The evaluation calculations are conducted before a definite period of time from which the scheduled traffic demands run. For this period, the service calculate to evaluate a significant number of combinations of paths and discover a suboptimal solution that can reduce traffic congestion. This calculation also runs when the service detects a gain in the packet loss rate between itself and another adjacent AR

1. Each node measures and sends network information to control service on the cloud computer network periodically.



2. Before scheduled traffic demands run or when packet loss rate runs over threshold, control service evaluates path combinations and searches better detour route.



3. New routing table are announced to each node when control service finds better path combinations. Then each node route packets according to new path combinations.

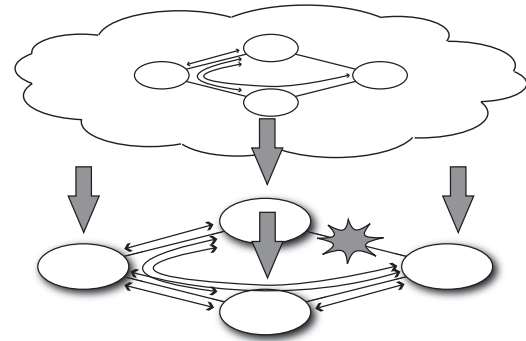


FIGURE 3.2: Rerouting for Overlay Network.

router. The procedure used for collecting the network information while changing the routing tables is shown in Fig. 3.2 and is enumerated below.

1. The AR routers in each facility send network information measured to the control service on a cloud computing environment on the Internet.
2. Before the definite period of time from when the time scheduled traffic demands run, or when the service detects a gain in the packet loss rate, the service starts to calculate the evaluation of path combinations. The service searches for a more suboptimal path combination that can reduce the packet loss rate.
3. When the service finds a combination of paths that improve the network conditions, the service announces new routing tables to all AR routers. After all AR routers receive the routing tables, they route according to these new routing tables.

### 3.3 Evaluation Calculation for a Combination of Paths

#### 3.3.1 Collection of Network Information

The AR routers in each facility collect the following four types of network information.

- One-way delay time between itself and any adjacent AR routers  $\delta$
- Packet loss rate between itself and any adjacent AR routers  $\lambda$
- Available bandwidth between itself and any adjacent AR routers  $\beta$
- Amount of traffic demands toward other AR routers  $\tau$

When the number of facilities is  $n$ , each facility is presented as  $N_1, \dots, N_n$ . An AR router in facility  $N_i$  sends probe packets towards an AR router in an adjacent facility  $N_j$ . Through these probe packets, AR routers measure the one way delay time  $\delta_{ij}$  between  $N_i$  and  $N_j$ . In addition, using this probe packet, the AR router in  $N_i$  estimates the available bandwidth  $\beta_{ij}$  between  $N_i$  and  $N_j$ . At  $N_i$ , the time stamp,  $t_{send}$ , when the packet is sent is described in the packet payload. When the packet reaches  $N_j$ , a time stamp,  $t_{recv}$ , is added to its payload when the packet arrives. This time information, i.e., both  $t_{send}$  and  $t_{recv}$ , are capsulated into the payload of the probe packet from  $N_j$  to  $N_i$ . When this packet reaches  $N_i$ , time information i.e., both  $t_{send}$  and  $t_{recv}$  is decapsulated, and the AR router obtains a one-way delay time,  $\delta_{ij} = t_{recv} - t_{send}$ . A probe packet has a sequential number based on each combination between itself and an adjacent facility. If the time information is not sent back for a definite period of time,  $d_{limit}$ , after a probe packet is sent, an AR router considers the probe packet as lost. From this lost information, AR routers measure the packet loss rate per unit of time (one second) for any logical connection line. To measure the available bandwidth, we propose the use of pathChirp [60] and an approach for approximating the cross traffic with fluidity [61] to consider the impact of short-term traffic variations and the required quality of movie delivering. There is more than one logical connection line between  $N_i$  and  $N_j$ . AR routers at  $N_i$  and  $N_j$  measure the one-way delay, available bandwidth and packet-loss rate of each line. AR routers distinguish each line individually. An AR router at  $N_i$  also measures the amount of traffic demands toward all facilities but itself, i.e.,  $\tau_{i1}, \tau_{i2}, \dots, \tau_{in} (n \neq i)$ . Probe packets are sent to only adjacent facilities at regular time intervals and routed by lower layer protocols. AR routers send the network information collected to the control service in a cloud computing environment (Fig. 3.3).

The clock synchronization of each AR router is necessary to measure a one-way delay. An accuracy of clock synchronization of within 100 ms is needed for this TE. An approach

for a precise estimation of the time-transfer is useful in the case of network congestion to realize this level of accuracy [50]. The amount of traffic from the probe packets is about 10 kbps if every AR router sends probe packets at 10 ms intervals. The probe traffic can be negligible for other network traffic demands.

### 3.3.2 Control Service

A control service in a cloud computing environment accumulates network information, manages the schedule of reserved traffic demands and calculates the evaluation value of combinations of the path combinations. The service has a queue with a length of  $w$  for each one-way delay time  $\delta$ , packet-loss rate  $\lambda$ , and available bandwidth  $\beta$  between each  $N_i$  and all of its neighbors  $N_j$ . There is also a queue for the amount of traffic demand  $\tau$  between each  $N_i$  and  $N$  other than  $N_i$ . When new network information reaches the service, the information is pushed to the head of each queue, and the oldest information is dropped. The queues always have the latest  $w$  network information.

Before a definite period of time from time which the scheduled traffic demands run, the control service calculates the evaluation value of the combinations of paths using the new network parameters altered based on the scheduled traffic demands. In searching for suboptimal combinations of paths, to increase the response performance for dynamic changes of the network environment, we not only adopt an algorithm for searching the optimal combination of paths but also adopt an incremental search algorithm that applies new routing tables once better combinations of paths improving the packet loss ratio above the smaller definite threshold level are found. The service therefore periodically announces a new routing table to every AR router, which can improve the packet loss ratio above a certain threshold. This evaluation calculation is also started by a triggering of the detected gain of the packet loss rate.

It takes a significant amount of time to calculate the evaluation values for all combinations of paths in a large-scale network. The calculation is stopped after a definite period of time. Although the network information continues to change while the evaluation values are calculated, the values are calculated based on the network parameters when a calculation is started. The parameters are not changed following network changes during the calculation. Although there are no scheduled traffic demands, or a gain in packet-loss rate, the combination of paths may become obsolete and may be far from optimal by changes in network conditions, such as changes in available bandwidth caused by network traffic from or to facilities not included in this overlay network. To break out from this obsolete local minimum combination, the evaluation values of the paths

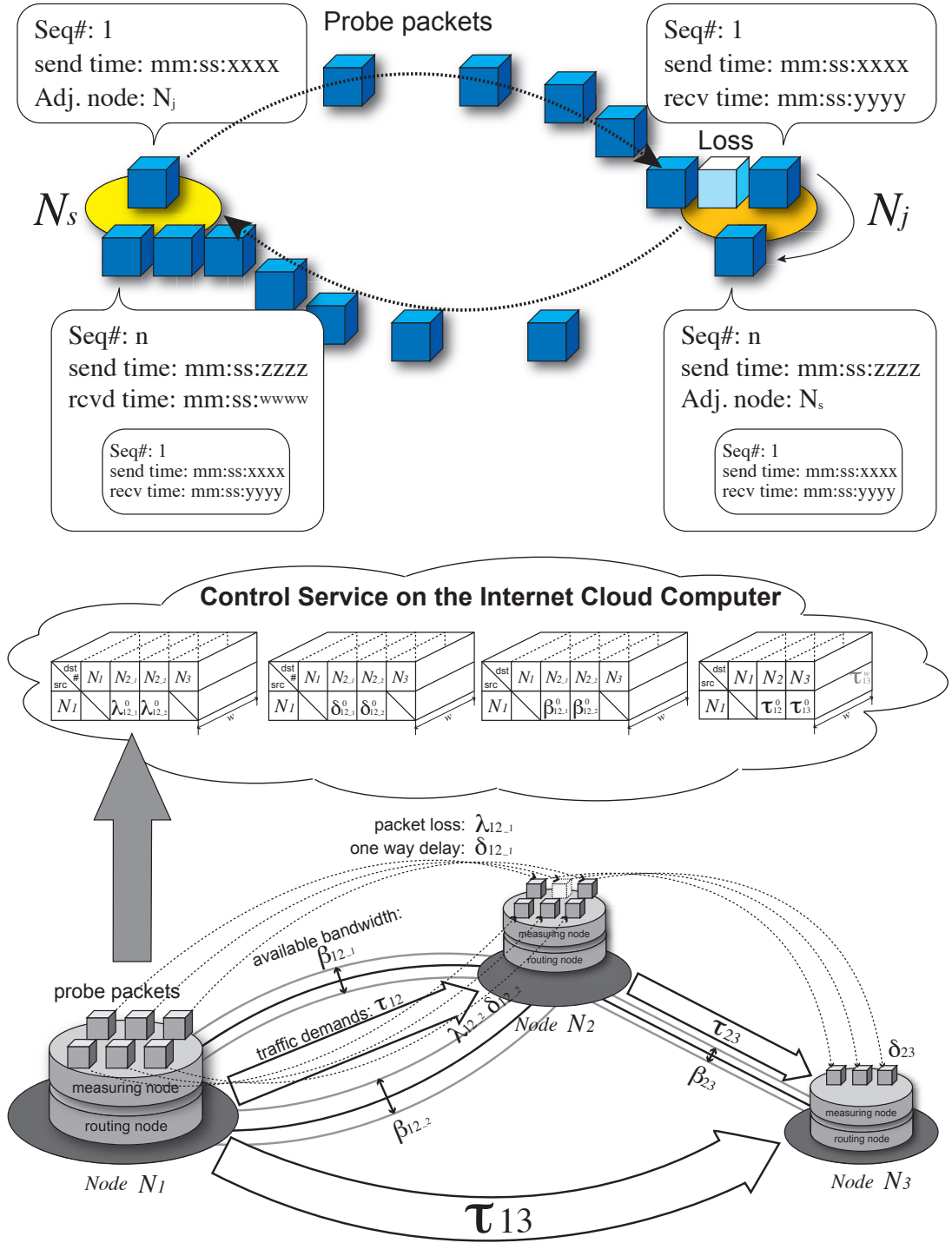


FIGURE 3.3: Measuring of Network Information.

combinations with the latest network information are calculated at regular time intervals. Through such calculations, if the combination which improves the above definite threshold level is found, the service announces a new routing table derived from the combination. The frequency of such announcements is longer than the maximum latency time between every AR router and service. By setting the time longer than the maximum latency time, every AR router can synchronize the timing to alter the new routing table. Through this synchronization, the packet disorder caused by a change in the routing table can be reduced.

Candidate path combinations are all combinations of paths without loops between each source node (src) and all destination node (dst). Outbound and inbound paths between dst and src are equal paths. One path is selected for every couple of dst and src pairs. A combination of paths is the superposition of every path for every couple of different node pairs. The amount of traffic demands for every couple is known through measurements by the AR routers. The evaluation value of a path combination is the estimated amount of packet loss and the total latency time calculated using a discrete event simulation with the measured network information.

### 3.3.3 Scoring of Paths

All paths between one dst and another src have their own score according to the path latency. Paths with a smaller score are selected into candidate combinations of paths more preferentially. The latency of a path is expressed by a summation of the one-way delay of each link included in the path.  $M_{sd}$  is the minimum latency of the path between src: $N_s$  and dst: $N_d$ . The score of the paths between  $N_s$  and  $N_d$  is expressed based on the latency of each path divided by  $M_{sd}$ . Thus, the score of the minimum latency path between  $N_s$  and  $N_d$  is certainly 1.0.

For the example four-node (facilities) ring topology shown in Fig. 3.4, in a couple of node pairs  $[N_1, N_2]$ , there are only two paths without any loops ( $\langle N_1, N_2 \rangle$  and  $\langle N_1, N_3, N_4, N_2 \rangle$ ). The total latency of path  $\langle N_1, N_2 \rangle$  is 2 ms. On the other hand, the total latency of path  $\langle N_1, N_3, N_4, N_2 \rangle$  is 18 ms. The minimum latency of  $M_{N_1 N_2}$  is 2 ms, the score of path  $\langle N_1, N_2 \rangle$  is 1.0, and the score of path  $\langle N_1, N_3, N_4, N_2 \rangle$  is 9.0 (18.0/2.0). The path score for the other node pair, i.e.,  $[N_1, N_3]$ ,  $[N_1, N_4]$ ,  $[N_2, N_3]$ ,  $[N_2, N_4]$ , and  $[N_3, N_4]$  is also calculated in a similar manner.

Combinations of paths with smaller score paths are made. The evaluation value is then calculated for a combination of paths. In Fig. 3.4,  $\langle 1.0, 1.5, 2.0, 2.3, 4.0, 9.0 \rangle$  is a list of sorted scores in ascending order. Following this order, a path that has each score in this list is selected into a set of path candidates, and every possible combination of paths

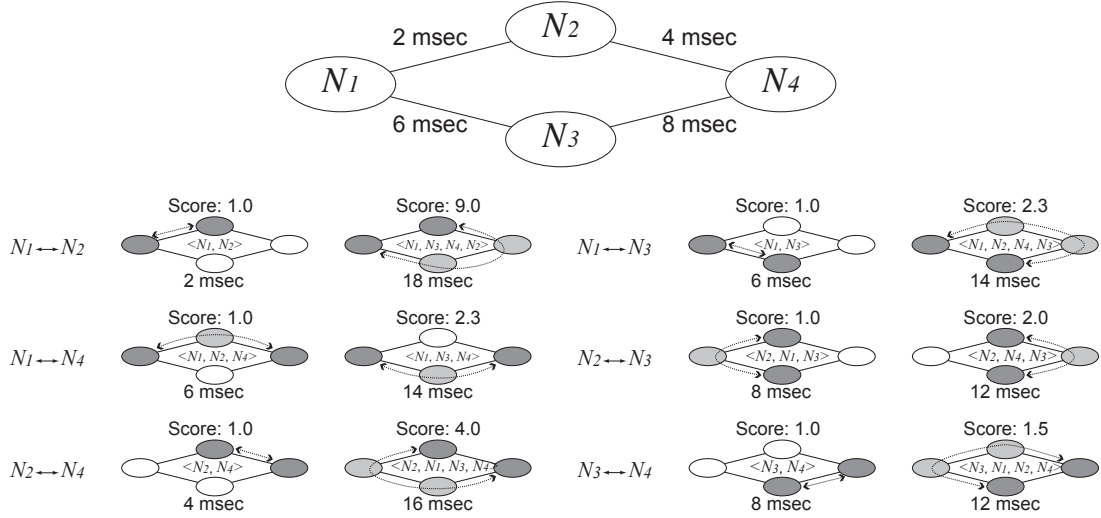


FIGURE 3.4: Path Scoring Based on Total Latency.

without any overlap is made according to this set of candidates. If there is more than one path with the same score, a path that has fewer hops is preferentially selected. In the first stage, the first combination of paths consists of every couple of two node with the minimum latency time because all of these paths have a score of 1.0. In Fig. 3.4, the first combination is  $[\langle N_1, N_2 \rangle, \langle N_1, N_3 \rangle, \langle N_1, N_2, N_4 \rangle, \langle N_2, N_1, N_3 \rangle, \langle N_2, N_4 \rangle, \langle N_3, N_4 \rangle]$ .

At the next stage, path  $\langle N_3, N_1, N_2, N_4 \rangle$  which has a score 1.5, is added to the set of candidates, and every possible combination of paths is made. Owing to the deduplication combinations, the newly created combinations of paths are only  $[\langle N_1, N_2 \rangle, \langle N_1, N_3 \rangle, \langle N_1, N_2, N_4 \rangle, \langle N_2, N_1, N_3 \rangle, \langle N_2, N_4 \rangle, \langle N_3, N_1, N_2, N_4 \rangle]$ .

At the even next stage, path  $\langle N_2, N_4, N_3 \rangle$  which has a score of 2.0, is added into the set. The newly-created combinations of paths are  $[\langle N_1, N_2 \rangle, \langle N_1, N_3 \rangle, \langle N_1, N_2, N_4 \rangle, \langle N_2, N_4, N_3 \rangle, \langle N_2, N_4 \rangle, \langle N_3, N_4 \rangle]$  and  $[\langle N_1, N_2 \rangle, \langle N_1, N_3 \rangle, \langle N_1, N_2, N_4 \rangle, \langle N_2, N_4, N_3 \rangle, \langle N_2, N_4 \rangle, \langle N_3, N_1, N_2, N_4 \rangle]$ . In the same way, paths  $\langle N_1, N_3, N_4 \rangle, \langle N_1, N_2, N_4, N_3 \rangle, \langle N_2, N_1, N_3, N_4 \rangle, \langle N_1, N_3, N_4, N_2 \rangle$  are added into the set, and every possible combination of paths is made, and duplicated combinations are removed.

By preferentially combining paths with paths with smaller scores, any suboptimal solutions with a combination of paths with a smaller total latency than the combination already evaluated is prevented from being found later. By selecting paths with a smaller hop number when more than one path have the same score, paths with less overlapping are preferentially combined. Combinations with less overlapping paths are expected to experience less cross-traffic. In this algorithm, we use only the latency time for the scoring, although the available bandwidth can also be used. When the available bandwidth

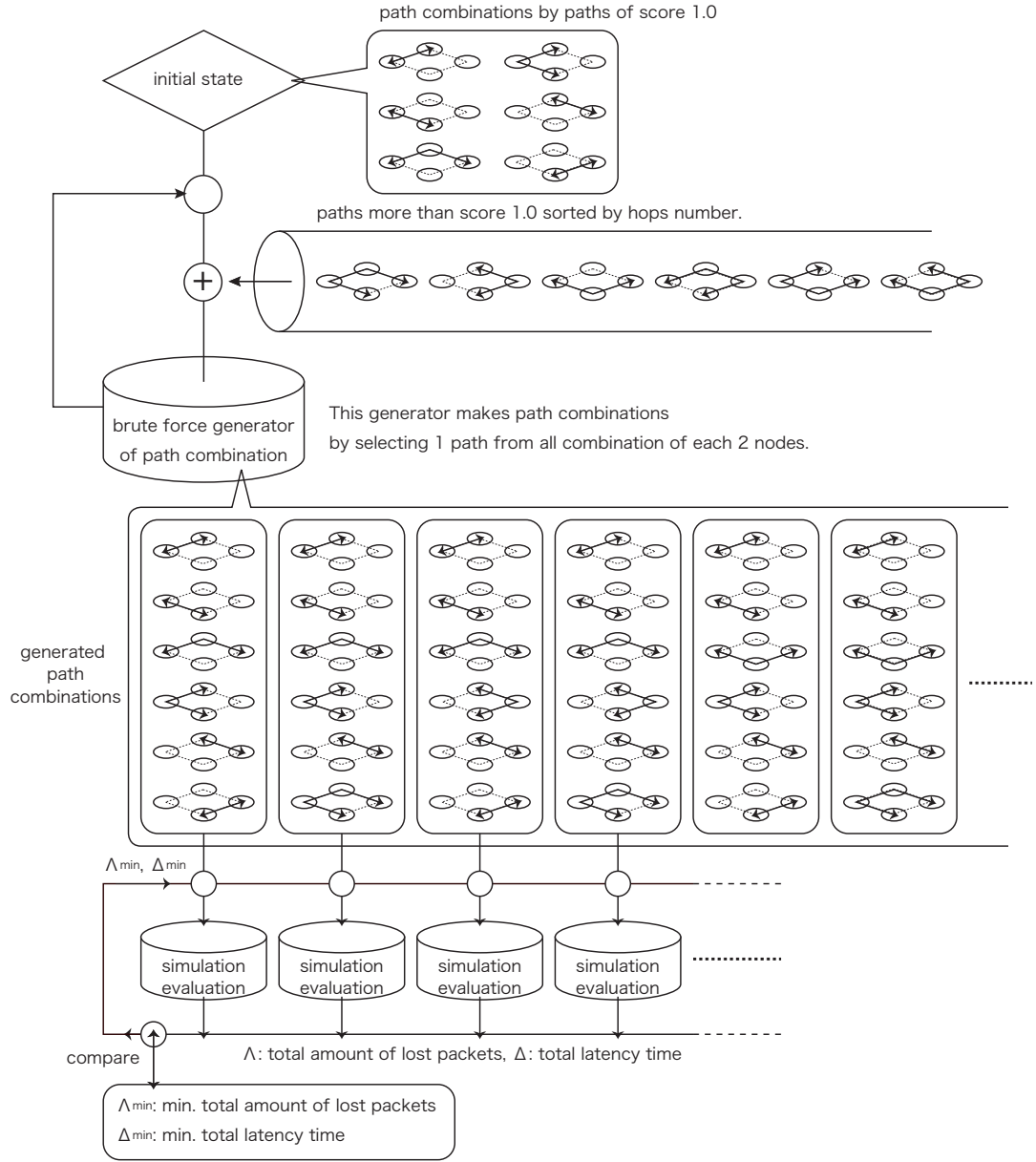


FIGURE 3.5: Making Combinations of paths and Evaluation.

is used, the score is calculated to weigh the summation of inverse available bandwidth by referencing the algorithm of OSPF [62].

### 3.3.4 Evaluation Calculation by Simulation

A routing table presented by a matrix table with src:  $N_s$  versus dst:  $N_d$  and the next hop  $N_n$  can be derived from a combination of paths. An evaluation value of each combination is calculated through a discrete event simulation. In the simulation, all nodes generate packets for a given length of time, and the simulation is run until all packets reach

their destination. Through the simulation, the total amount of packet loss  $\Lambda$  and total amount of latency time  $\Delta$  can be calculated. The network parameters of the simulation are based on the network information of the AR routers measured. In this algorithm, a network simulator is implemented by referencing the ns-2 implementation of packet queuing and packet loss [63], and is specialized to calculate the constant bit-rate traffic and routing pattern, the path of which is uniquely specified for every couple of node pairs.

When  $\Lambda$  calculated through simulation is less than the minimum  $\Lambda_{min}$  calculated through a past simulation,  $\Lambda_{min}$  and  $\Delta_{min}$  are altered to  $\Lambda$  and  $\Delta$ . The combination of paths related to this simulation then become a new suboptimal solution (suboptimal combination). If  $\Lambda$  is the same as  $\Lambda_{min}$ , the combination of paths that have a smaller total latency become a new suboptimal solution by comparing  $\Delta$  to  $\Delta_{min}$  (Fig. 3.5).

If  $\Lambda$  exceeds  $\Lambda_{min}$  during the calculation of a simulation, the calculation is aborted and a new simulation of the next path combination is started.  $\Lambda$  and  $\Delta$ , derived from one combination of paths, are independent from the result of the simulation calculation of another combination. Aborting the calculation therefore has no effect on the later simulation results. This also means that the simulation evaluation can be parallelized through every combination.

## 3.4 Examinations

### 3.4.1 Accuracy Evaluations of the Simulator

To evaluate the accuracy of the discrete event simulator implemented, the simulator is compared to ns-2 with a three-networks topology, in which the number of nodes is three, as described by BRITE [16]. The three-network topology is shown in Fig. 3.6. We use Barabási-Albert and Waxman models for graph generation. In the Waxman model, we use random and heavy-tailed distributions as the method of node placement because the structure of the graph is changed based on the nodes placement method used.

The bandwidth of each link included in these networks is 100 Mbps. The one-way delay of each link has four simulation steps (ss) on average with a uniform distribution. One simulation step corresponds to 1 ms. In delivering a movie on a computer network, traffic situations in which all nodes request traffic from every other node rarely occur. For an evaluate under a more realistic situation, we created 200 traffic patterns in which all nodes request traffic from three nodes on average based on referencing examples of movie delivery over the past ten years. The amount of traffic demand is  $2^n$  Mbps at a

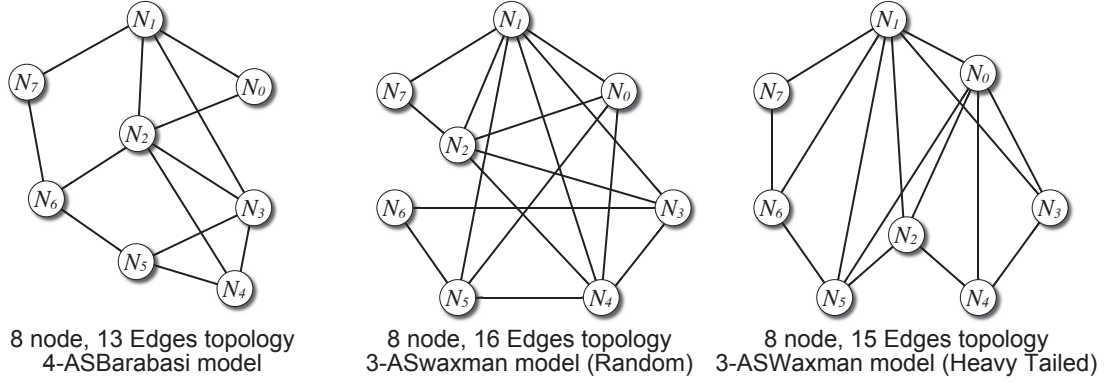


FIGURE 3.6: Networks for the Simulator Evaluation.

	4- ASBarabasi	3-ASWaxman (random)	3-ASWaxman (Heavy Tailed)
Total amount of Packet Loss	99.8% ( $\sigma = 2.13 \times 10^{-2}\%$ )	99.0% ( $\sigma = 2.02 \times 10^{-2}\%$ )	98.8% ( $\sigma = 2.00 \times 10^{-2}\%$ )
Total Latency Time	98.6% ( $\sigma = 5.88 \times 10^{-3}\%$ )	98.4% ( $\sigma = 7.17 \times 10^{-3}\%$ )	99.4% ( $\sigma = 5.61 \times 10^{-3}\%$ )

TABLE 3.1: Relative Precision of the Network Simulator to ns-2.

constant bit rate (where  $n$  is a distribution during  $0 \leq n \leq 4$ ). From the start of the simulation ( $ss = 0$ ) to 100 ss, every node runs a packet generation according to a the traffic pattern in each network topology. The end of the simulation is defined as the moment when all packets reach their destination.

To compare the implmented simulator and the ns-2, the total amount of packet loss and total latency time were calculated by both. Table 3.1 shows the results of the comparison. The ns-2 was configured to route the packets according to the specified path combination. The values in table 3.1 are the relative values of our simulator divided by the ns-2 results. Our simulator can calculate the total amount of packet loss and total latency time for an eight-node topology network and the traffic patterns with high accuracy.

### 3.4.2 Computation Time Evaluations of Simulator

To evaluate the computation time of our simulator, all combinations of paths are calculated to search the optimal solution on the 200 traffic patterns we created and for each network topology. Figs. 3.7 and 3.8 show the progress in the calculations and the

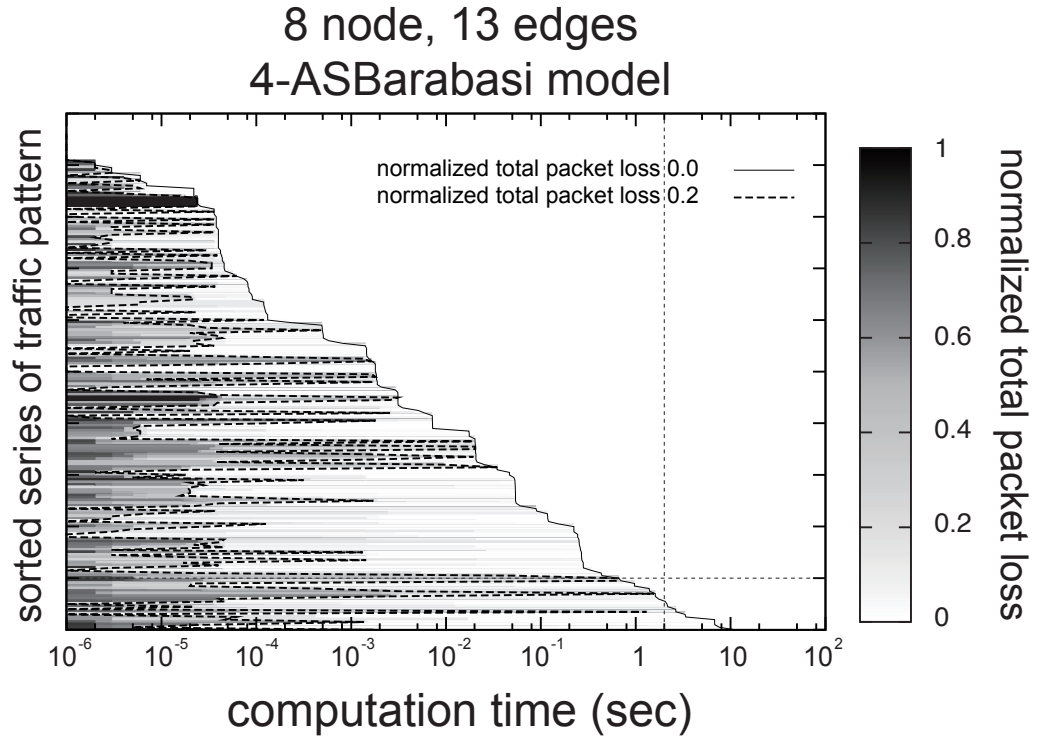


FIGURE 3.7: Convergence to Optimal Solutions in the Barabási-Albert Model Topology.

changes in the suboptimal solution discovered. The horizontal axis indicates the logarithmic time of the calculation and the vertical axis shows 200 traffic patterns sorted by ascending sequence of the total time required to discover the optimal solution.

For each traffic pattern, the total amount of packet loss is normalized based on the total amount for the optimal solution. A change in the normalized amount is presented through a gray-scale gradation. Points in which the optimal solution is discovered (equal to a point in which the normalized amount is 0) in each traffic pattern are connected by a solid line. Points in which the normalized amount is 0.2 are also connected by a solid line. This simulation of path combinations was executed on ten PCs with two Intel Xeon (2.93 GHz) CPUs, each with four cores and eight threads, and 2 GB of memory. The simulator was implemented in C++, and run on FreeBSD 8.2.

In all traffic patterns on each network topology, the simulator discovered the optimal solution within 11 s, and solutions of 0.2 normalized amounts within 2 s. In less than 10 % of the traffic patterns, the time required to discover the optimal solution exceeds 1 s.

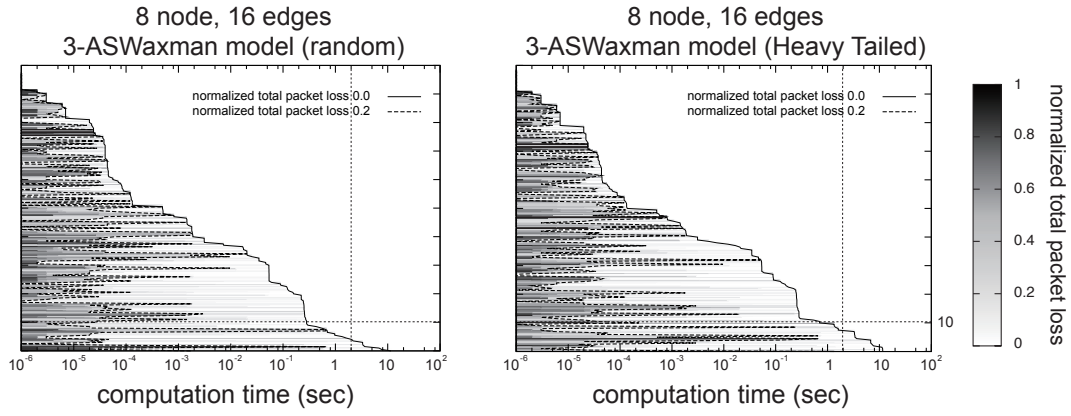


FIGURE 3.8: Convergence to Optimal Solutions in the Waxman Model Topologies.

### 3.4.3 Evaluation Examination

To show the advantage of the proposed method for a real network operation, by referencing the topology of past RIBB movie delivery trials, we created an eleven-node topology network with several detour paths (Fig. 3.9), and traffic patterns that incur congestion in several paths. We evaluated the proposed method using this topology and traffic patterns.

In this topology, links between neighboring nodes are full duplex links. The bandwidth and one-way delay are shown in Fig. 3.9. The network simulation was run on ns-2 and the combinations of paths and routing tables were calculated using the proposed method. A recalculation of the path combinations was run when the packet loss rate in any link exceeded 0.1 %. A probe packet was sent every 1 ms and the advertisements of the new routing tables have a minimum interval of 10 ms. According to the traffic patterns, all nodes generated packets for a given length of time. The simulation was ended when all packets reached their destination. We observed and evaluated changes in the total amount of packet loss and total latency time.

To evaluate the adaptability of the proposed method for network changes, we set up the network events shown below.

1. At 0.0 s (initial state), every node requests  $2^n$  Mbps traffic with a constant bit rate from all other nodes. Here,  $n$  is set to a uniform distribution during  $0 \leq n \leq 8$ .
2. From the elapsed time of 2.0 to 6.0 s, 512 Mbps traffic is scheduled from node #0 to node #10.

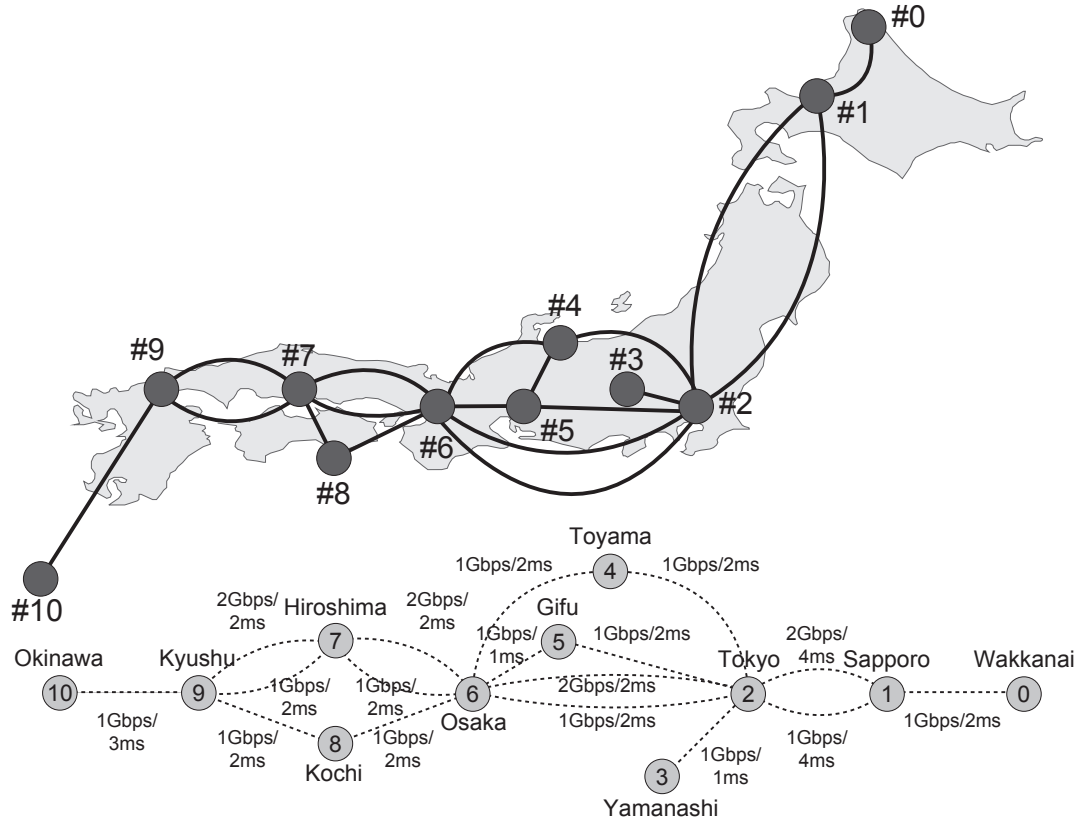


FIGURE 3.9: Network for Evaluation Experiment

3. During an elapsed time of 6.0 s, every node returns to the initial state, requests  $2^n$  Mbps traffic of a constant bit rate to all other nodes, where  $n$  is set to a uniform distribution during  $0 \leq n \leq 8$ .
4. From an elapsed time of 8.0 to 12.0 s, 512 Mbps of traffic is requested from node #0 to node #10. This request is not scheduled.

As with the evaluation described in section 3.4.2, this path combination simulation was executed on ten PCs with two Intel Xeon CPUs and 2 GB of installed memory. The total amount of packet loss on a congestion path is shown in Fig. 3.10.

Owing to a scheduled traffic request at an elapsed time of 2.0 s, it is necessary to calculate the evaluation of path combinations with a sufficient calculation time when enabling TE. Under this situation, the calculation is started from an elapsed time of 0.0 s, and at an elapsed time 1.68 s, the simulator finds the suboptimal solution in which there are no links with a packet loss. New routing tables are then derived and announced. When enabling TE, there are no traffic losses because all nodes are changed to the new routing table by the elapsed time of 2.0 s. On the other hand, at an elapsed time 8.0 s, packets are lost even when TE is enabled because these traffic demands are not scheduled and

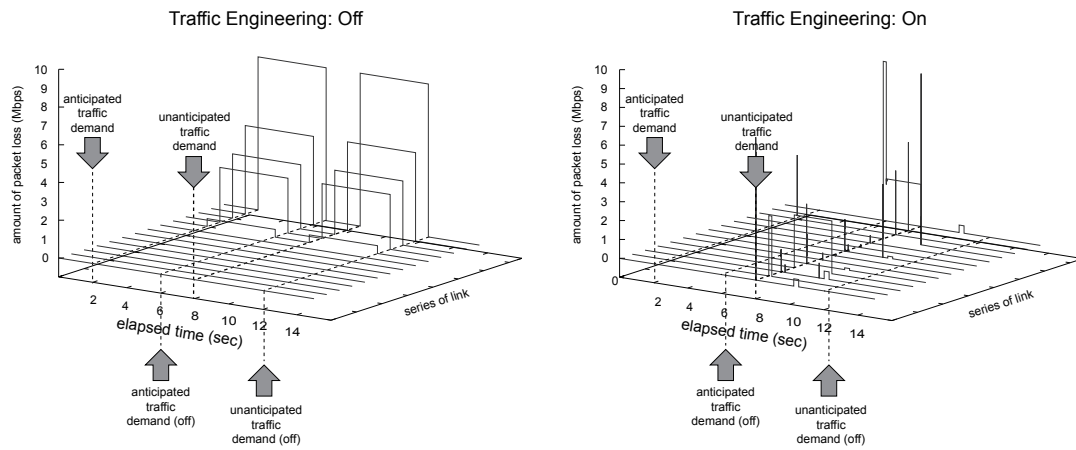


FIGURE 3.10: Change of Packet Loss.

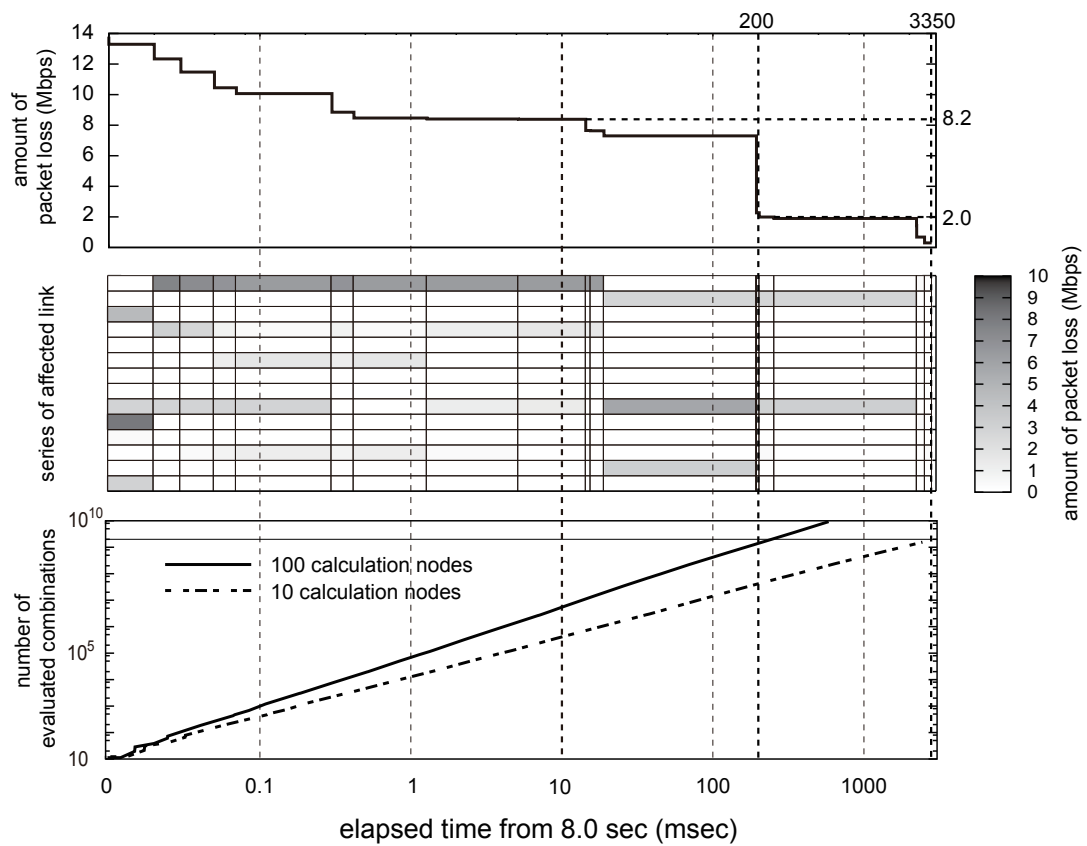


FIGURE 3.11: Change of Packet Loss and Evaluation for Combination of Paths.

an evaluation calculation for a detour route cannot be executed beforehand. AR routers notify a gain of packet loss and send this information to the control service. The control service then immediately starts evaluating the combinations of paths.

From an elapsed time 8.0 s, the detailed changes in total packet loss on several links in which packet loss occurs are shown in Fig. 3.11. The horizontal axis indicates the logarithmic elapsed time from an alteration of the routing table, which frequently occurs within 1.0 s from the start of the calculation when searching for a suboptimal solution. The total amount of packet loss is shown in the graph at the upper-side of the figure. Changes in packet loss from an elapsed time of 8.0 to 12.0 s are shown in the graph in the middle of the figure.

Though suboptimal solutions are discovered within an order of 0.1 ms, new routing tables are announced within an order of 10 ms. Within 10 ms after an unscheduled traffic request is run, a new routing table is announced and altered to new combinations of paths in which the total amount of packet loss is 8.2 Mbps. After several new routing tables are announced, within an elapsed time of 200 ms after an unscheduled traffic request is run, the routing tables are altered into new combinations of paths in which the total amount of packet loss is 2.0 Mbps. Finally, within an elapsed time of 3,350 ms, the routing tables are altered into path combinations without any packet losses.

The change in the number of calculated combinations is shown in the graph at the bottom of the figure. A real change using ten PCs and an estimated change from 100 PCs are shown. The total number of combinations of paths is  $1.7 \times 10^{58}$  in this topology.  $4.1 \times 10^9$  combinations are evaluated by the combination without any traffic loss is found in elapsed time 3,350 ms with 10 PCs.

## Chapter 4

# Implementation and Discussion

### 4.1 Discussion Regarding the Performance

#### 4.1.1 Decentralized Stochastic Approach

In the section 2.4 experiment 1. in which autonomous load balance features were evaluated at different increments of traffic demands, we compared the results when enabling AR features to those when disabling AR features. In a simple and easy topology, such as square in which a source node and a destination node are connected by two paths, it is common to distribute the traffic demands using an Equal Cost Multi-Path (ECMP). In ECMP, although an algorithm to separate the traffic demands depends on the implementation, separation on a per-flow basis highly recommended [64]. The large granularity of network flows such as a high-definition movie transmission cannot make efficient use of the paths, but when the granularity is sufficiently small, the traffic demands can be separated into the paths equally. In experiment 1., we suppose that paths AR4-AR3-AR1 and AR4-AR2-AR1 apply ECMP, and variations in the total delay are thought to be constricted to a small value when a large amount of traffic demand arises because we can estimate that there are no lost packets until the amount of the traffic demand are increases to 6 Mbps (3 Mbps on each path). On the other hand, the traffic demands are separated and become imbalanced with AR when using the proposed algorithm because the ratio of distribution depends on the delay time in each path, and thus the traffic distribution using ECMP will be effective under simple and uniform network conditions. To keep up with increment of traffic demands adaptively, it is necessary to introduce information on the bandwidth occupancy, available bandwidth, and amount of the traffic demand for a certain amount of packet loss into the packet distribution equation.

Through a deterministic approach such as ECMP, the traffic distribution can be efficient when the network topology and the traffic demand are given. However, when the topology is complex, it is difficult to select the paths. In turn, when the network nodes are very large, the traffic demands is never given. If all paths that have been assigned in advance change for the worse because of cross traffic and network congestion, it is necessary for ECMP operation to search for better paths and set them as paths with equal cost. However if there are no paths that have a sufficient amount of available bandwidth, it is difficult to distribute the traffic demands adaptively according to the available bandwidth. The proposed algorithm has an advantage in that it is not necessary for the algorithm to search for better paths and arrange the network manually because the algorithm determines the ratio of distribution autonomously according to the measured one-way delay.

In the section 2.4.5 experiment 2, we evaluated the ratio imbalance of distribution by purposely changing the delay time. Comparing Figure 2.27 to Figure 2.29, there is no transmission delay until the amount of traffic demands exceeds 3.0 Mbps, and when the amount exceeds 3.2 Mbps, a packet loss is observed. This is caused by 10 Base-T repeater switch. A switch has a very small buffer size and thus loses packets before the transmission is delayed. As a result, a penalty delay for a packet loss changes the ratio of distribution rather than the transmission delay. On the other hand, using a penalty delay, the algorithm can sensitively avoid a congestion of paths, releasing the congestion; the ratio of distribution then recovers to the ratio before the congestion by altering the scores in the routing table. Although the algorithm was shown to be effective in the network used for this evaluation experiment, greater adaptability and the ability to avoid congestion paths in a more complex network consisting of nodes with a large buffer size and under more complex traffic demand are required.

The algorithm determines the ratio of distribution using one-way delay information as an evaluation value. If there are alternative paths that have a long latency, such as a satellite connection or international detour, these paths are never used because the ratio of distribution is rounded to the nearest 10 % in the implementation. If there are two paths in which the latency of one path is 2.11-times longer than the other path with parameter  $\lambda = 4.0$ , the longer latency path is never selected in the implementation. In other words, we can control the oscillation of paths by setting parameter  $\lambda$ .

#### 4.1.2 Centralized Deterministic Approach

When the change in traffic is moderate, sufficient time to calculate the evaluation of path combinations can be prepared by detecting the gain in packet loss rate to alter

the new routing tables, such as in the simulation evaluation described in section 3.4.3. In another case in which a change in traffic is drastic, an altered routing table can not be a useful way to avoid network disorder. Under a drastic change in network traffic, the network parameters used in the simulation evaluation should be updated from the information measured by an AR router, and a new evaluation should be restarted even if the calculation of an earlier evaluation continues. It is necessary for the control service to calculate more path combinations under such drastic network changes. To obtain a more useful suboptimal solution quickly, it is useful to deploy VMs and add them for the computation node temporarily.

Because the evaluation described in section 3.4.3 was conducted using an ns-2, all node clocks are synchronized. In a real network, the nodes clocks each differ. Network problems caused by packet ordering and packet dropping can occur by difference in timing of the altered routing tables. To reduce these problems, a network application using an overlay network adds information of the error correction code and decodes it using a buffer. Altering the routing table much more frequently causes network instability. By setting the improved threshold of the altered routing table to a bigger value, network stability can be improved.

At the moment when the nodes alter the routing table, packets can be dropped because they are transferred on the overlay network using an older routing table, and the next hop cannot be defined in a node by the altered routing tables. This problem increases the packet losses and causes the overlay network to be transiently unstable. This problem is solved by using  $2^n$  routing tables in each node. These routing tables have their own generation number  $0 \sim 2^n - 1$  for identification, and the packets transferred on the overlay network must be described using the generated number that accompanies its  $n$  bits header. Using this identification number of the routing table, every packet is transferred using an adequate routing table in which the source node is defined even at the moment with the routing tables are altered. The Value of  $n$  is determined depending on the frequency of routing table alterations. In the case of altering the routing table at 1.0 s intervals, and when the total latency of a path that has the maximum total latency in the network is less than 1.0 s, an adequate value of  $n$  is 1.0 because the routing table cannot be altered more than twice while transferring one packet.

## 4.2 Implementation on OpenFlow

### 4.2.1 History of OpenFlow

Version 1.1 of the OpenFlow protocol was released on February 28, 2011<sup>1</sup>, and the new development of the standard is managed by the Open Networking Foundation (ONF). In December 2011, the ONF board approved OpenFlow version 1.2<sup>2</sup>, and published it in February, 2012. The current version of OpenFlow is 1.4.0<sup>3</sup>.

In May 2011, Indiana University launched its SDN Interoperability Lab<sup>4</sup> in conjunction with the Open Networking Foundation to test how well the Software-Defined Networking and OpenFlow products of different vendors work together. In February 2012, Big Switch Networks released Project Floodlight, OpenFlow Controlleran that is Apache-licensed open-source software, and announced its OpenFlow-based SDN Suite in November of that year, which contains a commercial controller, and virtual switching and tap monitoring applications.

In February 2012, HP stated that it is supporting the standard on sixteen of its Ethernet switch products. In April 2012, Google's Urs Hölzle described how the company's internal network had been completely re-designed over the previous two years to run under OpenFlow with substantial improvements in efficiency. In January 2013, NEC unveiled a virtual switch for Microsoft's Windows Server 2012 Hyper-V hypervisor, which was designed to bring OpenFlow-based software-defined networking and network virtualization to those Microsoft environments<sup>5</sup>.

### 4.2.2 Specifications of OpenFlow

A part of specifications of OpenFlow 1.3.2 is shown below.

An OpenFlow switch consists of one or more flow tables and a group table which perform packet lookups and forwarding, and an OpenFlow channel to an external controller (Figure 4.1). The switch communicates with the controller, and the controller manages the switch through the OpenFlow protocol.

Using the OpenFlow protocol, the controller can add, update, and delete flow entries in the flow tables, both reactively (in response to packets) and proactively. Each flow

---

<sup>1</sup><http://archive.openflow.org/documents/openflow-spec-v1.1.0.pdf>

<sup>2</sup><https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.2.pdf>

<sup>3</sup><https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.4.0.pdf>

<sup>4</sup><http://incntre.iu.edu/SDNlab>

<sup>5</sup><http://en.wikipedia.org/wiki/OpenFlow>

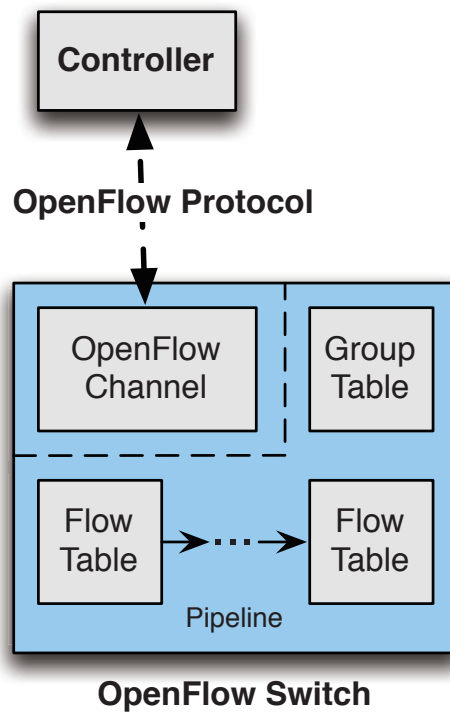


FIGURE 4.1: Main components of an OpenFlow switch.

table in a switch contains a set of flow entries; each flow entry consists of match fields, counters, and a set of instructions to apply to matching the packets.

Matching starts at the first flow table and may continue to additional flow tables. Flow entries match packets in priority order, with the first matching entry in each table being used. If a matching entry is found, the instructions associated with the specific flow entry are executed. If no match is found in a flow table, the outcome depends on the configuration of the table-miss flow entry; for example, the packet may be forwarded to the controller over the OpenFlow channel, dropped, or continued to the next flow table.

Instructions associated with each flow entry either contain actions or modify the pipeline processing. Actions included in the instructions describe the packet forwarding, packet modification, and group table processing. Pipeline processing instructions allow packets to be sent to subsequent tables for further processing and allow information, in the form of metadata, to be communicated between tables. Table pipeline processing stops when the instruction set associated with a matching flow entry does not specify the next table; at this point, the packet is usually modified and forwarded.

Flow entries may be forwarded to a port. This is usually a physical port, but it may also be a logical port defined by the switch, or a reserved port defined by this specification. Reserved ports may specify generic forwarding actions such as sending to the controller,

flooding, or forwarding using non- OpenFlow methods, such as normal switch processing, while switch-defined logical ports may specify link aggregation groups, tunnels, or loopback interfaces.

Actions associated with flow entries may also direct packets to a group, which specifies additional processing. Groups represent sets of actions for flooding, as well as more complex forwarding semantics (e.g., multipath, fast reroute, and link aggregation). As a general layer of indirection, groups also enable multiple flow entries to forward to a single identifier (e.g., IP forwarding to a common next hop). This abstraction allows common output actions across flow entries to be changed efficiently.

The group table contains group entries; each group entry contains a list of action buckets with specific semantics dependent on the group type. The actions in one or more action buckets are applied to packets sent to the group. Switch designers are free to implement the internals in any way that is convenient, provided that the correct match and instruction semantics are preserved. For example, while a flow entry may use all groups to forward to multiple ports, a switch designer may choose to implement this as a single bitmask within the hardware-forwarding table. Another example is matching; the pipeline exposed by an OpenFlow switch may be physically implemented with a different number of hardware tables.

OpenFlow-compliant switches come in two types: OpenFlow-only and OpenFlow-hybrid. OpenFlow-only switches support only OpenFlow operations, and in these switches, all packets are processed by the OpenFlow pipeline, and they cannot be processed otherwise.

OpenFlow-hybrid switches support both OpenFlow operations and normal Ethernet switching operations, i.e., traditional L2 Ethernet switching, VLAN isolation, L3 routing (IPv4 routing, IPv6 routing...), ACL, and QoS processing. These switches must provide a classification mechanism outside of OpenFlow that routes traffic to either the OpenFlow pipeline or a normal pipeline. For example, a switch may use a VLAN tag or input port of the packet to decide whether to process the packet using one pipeline or the other, or it may direct all packets to the OpenFlow pipeline. This classification mechanism is outside the scope of this specification. An OpenFlow-hybrid switch may also allow a packet to go from the OpenFlow pipeline to the normal pipeline through the NORMAL and FLOOD reserved ports.

The OpenFlow pipeline of every OpenFlow switch contains multiple flow tables, with each flow table containing multiple flow entries. The OpenFlow pipeline processing defines how packets interact with these flow tables. An OpenFlow switch is required to have at least one flow table, and can optionally have more. An OpenFlow switch

with only a single flow table is valid, and in this case, pipeline processing is significantly simplified.

The flow tables of an OpenFlow switch are sequentially numbered, starting at 0. Pipeline processing always starts at the first flow table; the packet is first matched against flow entries of flow table 0. Other flow tables may be used depending on the outcome of the match in the first table.

To select a flow entry, when processed by a flow table, the packet is matched against the flow entries of the flow table. If a flow entry is found, the instruction set included in that flow entry is executed. These instructions may explicitly direct the packet to another flow table (using a Goto instruction), where the same process is repeated again. A flow entry can only direct a packet to a flow table number greater than its own flow table number; in other words, pipeline processing can only go forward and not backward. Clearly, the flow entries of the last table of the pipeline cannot include a Goto instruction. If the matching flow entry does not direct packets to another flow table, the pipeline processing stops at this table. When the pipeline processing stops, the packet is processed with its associated action set and is usually forwarded.

If a packet does not match a flow entry in a flow table, it is considered a table miss. The behavior of a table miss depends on the table configuration. A table-miss flow entry in the flow table can specify how to process unmatched packets: the options include dropping them, passing them to another table, or sending them to the controller over the control channel using packet-in messages.

The OpenFlow pipeline and various OpenFlow operations process packets of a specific type in conformance with the specifications defined for that packet type, unless the present specifications or OpenFlow configuration specify otherwise. For example, the Ethernet header definition used by OpenFlow must conform to the IEEE specifications, and the TCP/IP header definition used by OpenFlow must conform to the RFC specifications. Additionally, packet reordering in an OpenFlow switch must conform to the requirements of the IEEE specifications, if the packets are processed by the same flow entries, group bucket, and meter band.

A flow table consists of flow entries. Each flow table entry contains the following:

**match fields** used to match against packets. These consist of an ingress port and packet headers, and optionally metadata specified by a previous table.

**priority** matching for the precedence of the flow entry.

**counters** for updating when packets are matched.

**instructions** used to modify the action set or pipeline processing.

**timeouts** where the maximum amount of time or an idle time before a flow is expired by the switch.

**cookie** which are opaque data values chosen by the controller, may be used by the controller to filter flow statistics, flow modification and flow deletion. They are not used when processing packets.

A flow table entry is identified by its match fields and priority; the match fields and priority taken together identify a unique flow entry in the flow table. The flow entry that wildcards all fields (all fields omitted) and has priority equal to 0 is called a table-miss flow entry.

### 4.3 Implementation of the decentralized stochastic approach on OpenFlow

As described in the section [2.4.3](#), AR of a decentralized stochastic approach for the experimental evaluation was implemented on a NetBSD 4.0.1 operating system. In the evaluation, traffic demands was distributed into the two paths using a per-packet routing method. In OpenFlow, the minimum unit on which OpenFlow can control is not a packet, but flow that is defined by the combination of source IP address, source port, destination IP address, and destination port. By the reason, per-packet routing method can not be adopted in the case to implement the decentralized stochastic approach using OpenFlow. In the approach, by using an xFlow architecture such as a sFlow, traffic demands can be distributed in the ratio calculated by the decentralized stochastic manner.

#### 4.3.1 xFlow and sFlow

xFlow is a generic name of a protocol for monitoring network, wireless and host devices. sFlow is one of a well-known protocol of xFlow. The sFlow.org consortium is the authoritative source for the sFlow protocol specifications. sFlow is currently at version 5. Previous versions of sFlow, including RFC 3176, are now obsolete. sFlow uses sampling to achieve scalability, and for this reason is applicable to high-speed networks (gigabits per second speeds and higher). sFlow is supported by multiple network device manufacturers and network management software vendors.

An sFlow system consists of multiple devices performing two types of sampling: random sampling of packets or application layer operations, and time-based sampling of the counters. The sampled packet/operation and counter information, referred to as flow samples and counter samples respectively, are sent as sFlow datagrams to a central server running software that analyzes and reports on network traffic, i.e., the sFlow collector. Based on a defined sampling rate, i.e., an average of 1 out of  $n$  packets/operations, is randomly sampled. This type of sampling does not provide 100% accurate, but does provide quantifiable accuracy. A polling interval defines how often the network device sends interface counters. sFlow counter sampling is more efficient than SNMP polling when monitoring a large number of interfaces. The sampled data are sent as a UDP packet to the specified host and port. The official port number for sFlow is port 6343. The lack of reliability in the UDP transport mechanism does not significantly affect the accuracy of the measurements obtained from an sFlow agent. If counter samples are lost, new values will be sent when the next polling interval has passed. The loss of packet flow samples results in a slight reduction of the effective sampling rate.

The UDP payload contains an sFlow datagram. Each datagram provides information about the sFlow version, the IP of the originating device, the sequence number, the number of samples it contains and one or more flow and/or counter samples.

### 4.3.2 Overview of the implementation

Each node consists of switches which supports xFlow protocol or OpenFlow. There is no matter whether the switches are implemented by hardware or software. Computers on which xFlow sampler and OpenFlow controller work are also necessary. There is also no matter whether the computers consist of hardware ones or software ones like virtual machines.

As a result of sampling the inbound traffic on a node, the xFlow sampler can obtain estimated amount of traffic demands for any destination IP address in IP addresses, source ports, and destination ports. On the other hand, each node creates the routing table of its own by information of feedback packets on the deterministic stochastic approach. Based on the routing table, this approach determine the distribution ratio of next adjacent nodes for any destination IP addresses.

If each node routes traffic demands in any flows, routed paths can be specified to one path for a combination of a source IP address, a source port and a destination port. Next node candidates can be also specified for the combination because candidate nodes that can cause a loop path are deleted. In any nodes, combinations for any next nodes can

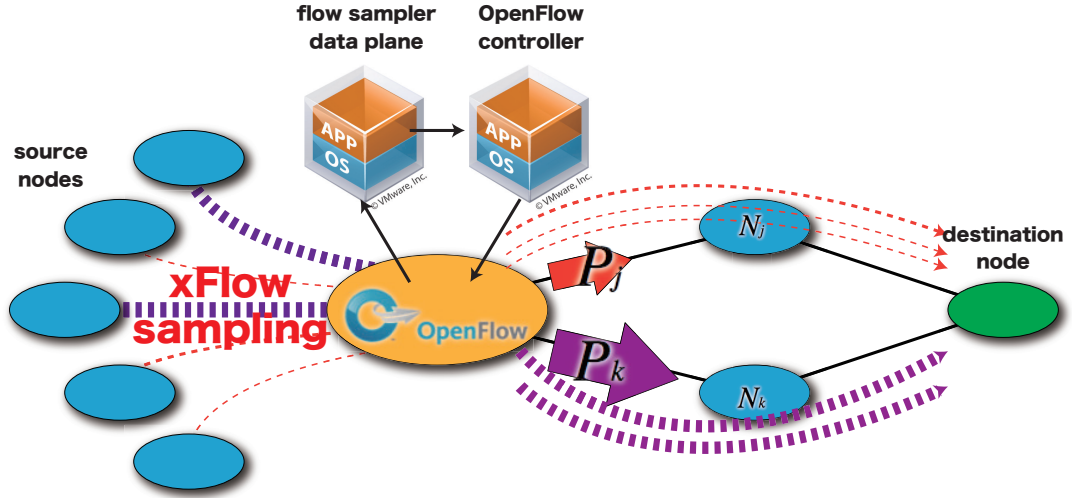


FIGURE 4.2: Overview of the implementation of the decentralized stochastic approach

be also specified to be assigned. When all combinations are listed that can be assigned, a several combination can be assigned to several next node candidates.

The amount of inbound traffic demands for any destination node can be estimated based on the result xFlow sampling. The distribution ratio can be convert to a distribution amount of traffic demands. And any amount of traffic demands for the combination of a source IP address, a source port, a destination IP address can be also estimated. The combinations are assigned to the next node candidate in a ascending order of the amount of traffic demands according to the distribution amount of traffic demands.

Depending on granularity of a deviation of size of traffic demands, traffic demands can not be distributed in an efficient precision. An implementation of equal cost multi paths (ECMP) based on OpenFlow is also suggested. The distribution can be efficient precision with dividing traffic demands using the implementation (Figure 4.2).

### 4.3.3 Procedure in the implementation

Each node has three types of controller. One is a xFlow sampler also known as a xFlow collector. One is a controller which create the routing table of the decentralized deterministic approach and determine the distribution ratio. And the other is a OpenFlow controller.

The xFlow collector on a node regularly communicates with xFlow switch of the node and sample the inbound flow. By the result of the sampling, the controller always maintain a table consisted of a source IP address, a source port, a destination IP address, and a

destination port. The controller also maintain the estimated amount of traffic demands of the four combination and the route of the combination.

The controller that determine the distribution ratio alters the routing table and distribution ratio to the next node candidates for any destination nodes. The controller communicate with the xFlow collector to obtain the information about the combination of a source IP address, a source port and a destination IP address for any destination IP addresses. The controller also obtain the amount of traffic demands of the combination and its path routed. A flow is defined as the combination.

The controller create a table of flow assignment according to the distribution ratio based on the flow information. When the amounts of traffic demands are constant, the tables are static. When the amounts of the demands change, the tables changes dynamic. To avoid frequent changes of the tables, the tables are altered when the traffic demand changes greater than a given threshold value. When the table is altered on the node, the controller send a signal to the OpenFlow controller.

When the OpenFlow controller received the signal of altering the table, the controller get the table of flow assignment from the controller that determine the distribution ratio. And the OpenFlow controller convert the table into a flow table formatted with the OpenFlow protocol. OpenFlow can controll the flow using the information of Layer 3 e.g., IP addresses and the information of Layer 4 e.g., port numbers. The flow table consists of a source IP address, a source port, a destination IP address, and a destination port. The OpenFlow controller connect the combination toward the physical port related to the specified physical port. The physical port does not mean a port number of Layer 4, but port of Layer 1 of the OpenFlow switch.

The flow table of OpenFlow is reflected to the OpenFlow switch using its API. By the reflection, the OpenFlow switch can controll the flow according to the deterministic stochastic approach (Figure 4.3).

## 4.4 Implementation of the centralized deterministic approach on OpenFlow

In a centralized deterministic approach, all combinations of node pairs are underspecified. Using the OpenFlow 1.3.2 specification, the flow entry is determined by a couple of source IP addresses or address range, and destination IP addresses or address range. This is the routing table of a centralized deterministic approach. In this approach, a routing table is the diagram of the next hop for the source node and destination nodes.

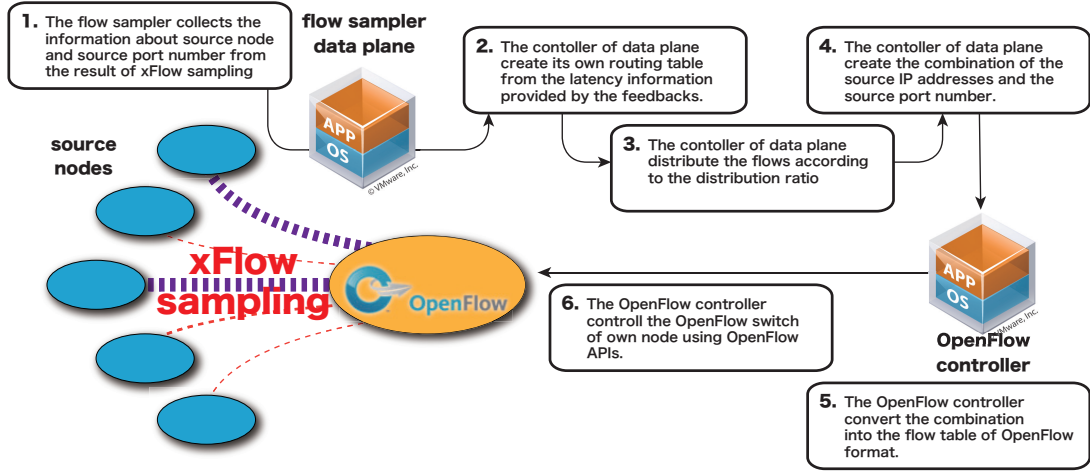


FIGURE 4.3: Procedure of the implementation of the decentralized stochastic approach

This coupling is equal to one flow table. An OpenFlow router can operate according to the description of all couples in the routing table for all nodes in the flow tables.

#### 4.4.1 Overview and procedure of the implementation

Based on the network information that each node collects, the controll service evaluate the combination of paths. When the service find out the combinatino that improve the evaluation values greater than the given threshold, the service create the routing table of each node according to the combination of paths. The service send the routing table to each node. Each node routes the flow based on the routing table. The routing table consists of a source IP address, destination IP address and a next node and a link ID for the combination of the IP addresses.

To select a flow entry, when processed by a flow table, a packet is matched against the flow entries of the flow table. If a flow entry is found, the instruction set included in that flow entry is executed. These instructions may explicitly direct the packet to another flow table (using the Goto instruction), where the same process is repeated again. A flow entry can only direct a packet to a flow table number greater than its own flow table number; in other words pipeline processing can only go forward and not backward. Obviously, the flow entries of the last table of the pipeline can not include the Goto instruction. If the matching flow entry does not direct packets to another flow table, pipeline processing stops at this table. When pipeline processing stops, the packet is processed with its associated action set and usually forwarded.

If a packet does not match a flow entry in a flow table, it is considered a table miss. The behavior of a table miss depends on the table configuration. A table-miss flow entry in

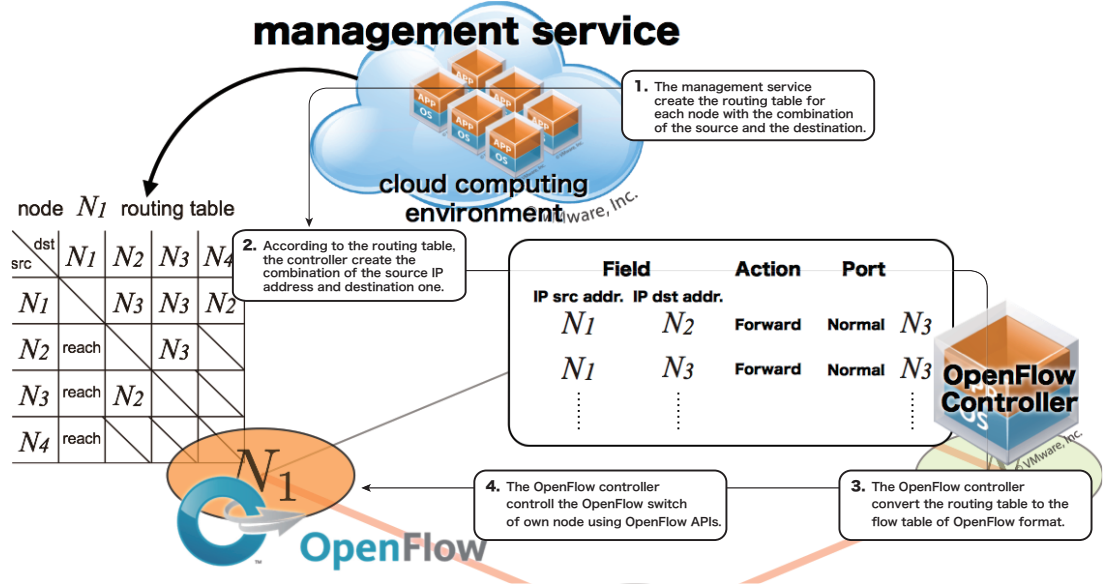


FIGURE 4.4: Procedure of the implementation of the centralized deterministic approach

the flow table can specify how to process unmatched packets; options include dropping them, passing them to another table or sending them to the controller over the control channel using packet-in messages.

The OpenFlow pipeline and various OpenFlow operations process packets of a specific type in conformance with the specifications defined for that packet type, unless the present specifications or OpenFlow configuration specify otherwise. For example, the Ethernet header definition used by OpenFlow must conform to the IEEE specifications, and the TCP/IP header definition used by OpenFlow must conform to the RFC specifications. Additionally, packet reordering in an OpenFlow switch must conform to the requirements of the IEEE specifications, provided that the packets are processed by the same flow entries, group bucket, and meter band (Figure 4.4).

## 4.5 Hybrid Approach

As mentioned before, the decentralized stochastic approach is rich in its scalability. Each AR router can participate the overlay network by only configuring the information about adjacent nodes and their links. Though a size of the routing table of each node increase on the order of  $O(n^2)$ , a record of each routing table is sufficiently small. For example, when the length of scores of each routing table,  $w$ , is 254, each score and a node ID is expressed by a integer of 24 bit, and an average number of adjacent nodes on each node is  $\frac{N}{3}$ , the size of routing table of each node with 2048 nodes network is 1GB.

On the other hand, jitter of total delay time has to be more than 0 except when  $\lambda \rightarrow \infty$  in the decentralized stochastic approach because of stochastic distribution of traffic demands. The jitter can grow better between a couple of nodes with large number of hops and long latency. When using penalty scores into score sequence for dealing with packet losses, the quality of network communication can go down by oscillation of distribution ratio of the traffic demands.

The centralized deterministic approach has an advantage to ensure the quality of the network communication. In the approach, the jitter is attributed to the jitter of routing on L3 network and switching on L2 network because the path between a couple of two nodes is specified. It is difficult for any other overlay routing algorithm to reduce the jitter than the proposed approach. As mentioned on the section 3.2, the approach succeeded to obtain the optimal solution of the paths combination on the eight-node topology network and 200 patterns of traffic by using 10 calculation nodes with 2.93 GHz, 12 cores in each node. It is shown that the proposed approach can deal with the network disorder rapidly enough.

On the other hand, elapsed time to obtain the optimal solution depend heavily on the resource of computation nodes. On the 11-nodes network shown in the section 3.3, the total number of paths combination is larger than  $5.3 \times 10^5$ . Even when using 10 calculation nodes with 2.93 GHz, 12 cores in each node, it takes a fair amount of time to evaluate the whole combination and find the optimal solution. In light of usage fee of several cloud computing services, five thousand dollars is needed to maintain VMs with 120 cores per a month As of 2014. It can be said that the proposed approach can be deal with the 8-nodes network at the current moment.

So it would appear that a hybrid approach that combine the advantages of two proposed approaches can have the better advantage. The decentralized stochastic approach has the better advantage on scalability although the approach is pointed out to have disadvantage on the jitter between a couple of two nodes with many hops, long latency. Meanwhile, the centralized deterministic approach has the better advantage on the quality of network communication with a high performance of searching the optimal solution on 8-nodes network and the approach can reduce the jitter than the stochastic approach. In the hybrid approach, nodes of the overlay networks are divided with clusters among which the maximum latency is less than given threshold. Inside of the clusters, the nodes route the flows with the decentralized stochastic approach. Between the clusters, the nodes route the flows with the centralized deterministic approach. In the evaluation experience shown in the section 2.3, the oscillation of the distribution ratio is occurred between a couple of the two nodes between that the latency is larger than 30 ms. In Japan, almost every latency between a couple of the capital cities is less than 30 ms. If

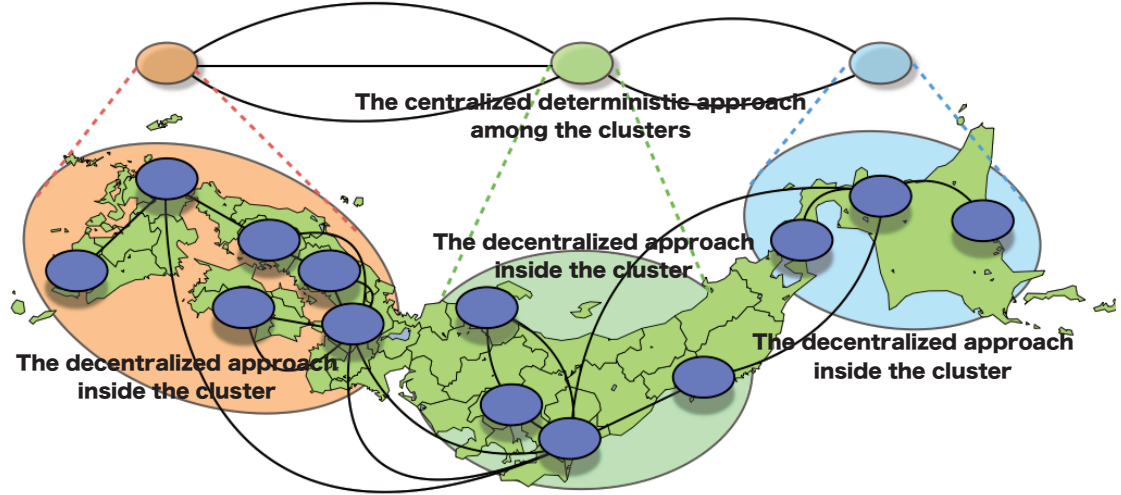


FIGURE 4.5: Overview of the hybrid approach

the given threshold is 30 ms, the nodes in Japan is included in the one cluster (Figure 4.5).

In the hybrid approach, both of the decentralized stochastic approach and the centralized deterministic approach are implemented in a AR. There is no need to get new hardware because the both approaches are available for implementation with OpenFlow protocol. When the controllers of each node works on the virtualization infrastructure on the node, the VMs can be deployed easily by using the VM appliances of the both approaches. The ARs can participate the overlay network by providing the information about the adjacent nodes and the network lines. When the ARs connect to the overlay network, ARs trade their topology information each other. The ARs measure the one way delay, the loss ratio, and the available bandwidth between own node and the adjacent nodes. The ARs also measure the amount of traffic demands between own node and every other node. The ARs send the information to the control service on the cloud computing environment.

The control service create the self-organizing map using the latency information and divide the topology into the clusters by the nodes among which the maximum latency is less than the threshold latency. The service send the information about a topology of the cluster to each AR. According to the information, boundary ARs are determined. The boundary ARs send the inter-cluster routing table to the nodes that are included in the cluster. Inside of the cluster, the flows are controlled by the decentralized stochastic approach.

The flow toward the destination node outside of the cluster, the flow is capsuled by the decentralized approach at first, and then the flow is re-capsuled with convert its

destination node to the boundary node of its cluster to the destination cluster. When the flow reaches the boundary node, the flow is decapsuled and sent to the port connected to the next cluster by the centralized approach.

## Chapter 5

# Conclusion

### 5.1 Summary

We proposed an adaptive routing algorithm REI in which a node works as a distributed autonomous agent. Scores of the routing paths are evaluated in terms of the latency, and they are shared by all nodes along the path. According to the scores obtained, every node independently selects a desirable next hop toward the final destination. REIsx is a type of REI, in which a time-invariant parameter,  $\lambda$ , specifies the degree of randomness in selecting the next hop.

We implemented a 64-node network simulator to evaluate our approach. Through simulation evaluations, REIsx can select shorter latency paths adaptively to the changes of the network traffic environment and deal with much more network traffic than a conventional routing algorithm such as OSPF.

It is an interesting problem to adjust parameter  $\lambda$  to the convergence of the total latency. A reduction in the size of the network routing table and a more realistic simulation evaluation are future works that we intend to carry out. We are planning to implement a self-assembly network that can autonomously partition the network routing domain.

In addition, we proposed a probabilistic routing method, NREI, which can adapt to dynamic changes in network traffic and the quality of network paths by autonomously measuring the one-way delay. Each routing node constructs its own routing table independently, and distributes inbound traffic demand toward shorter delay paths through weighted random selection. We set the routers implemented through this method at several research institutes connected to the testbed network, JGN-X; connected each router using a network connection line; and constructed an experimental network.

The information from a one-way delay measured by probe packets is reflected in the next node selection. The shorter latency paths are selected at a higher ratio. In this method, it is not necessary to input the information regarding each routing node. All routing nodes operate using a simple algorithm. When the network topology grows larger and more complex, there is no need for a manual arrangement according to changes in topology. The scalability and flexibility can be autonomously maintained. This also means that there is no centric routing information node or SPOF. The algorithm distributes the traffic demands to multiple paths and selects shorter total delay paths so that whole networks can afford a larger amount of traffic demand than a deterministic approach. Through evaluation experiments, we showed that the oscillation of the distribution ratio comes under control at below 7 %.

Because the algorithm determines the distribution ratio using only information from a one-way delay, adaptive routers always distribute the inbound traffic demands in a weighted random selection even if there are no packet losses or congestion paths. Though the average total delay is increased compared with the optimized solution for a small amount of traffic demand, the algorithm is effective, particularly under network conditions in which large amounts of traffic demand, such as high-definition movie transmissions, are consistently produced. A probabilistic spreading of the selected paths is controlled by parameter  $\lambda$ . When  $\lambda$  has a bigger value, the average total delay is decreased, but the dispersion of the total delay is increased. To set the proper value of  $\lambda$ , we balance the average and dispersion of the total delay. We can also reduce the maximum value of the total delay using this parameter.

In this implementation, the quality of the connection can be lost because of the per-packet routing. As future work, we intend to improve the algorithm using more network information such as the available bandwidth and network availability to achieve both a packet processing ability and high quality network communication.

Finally, we proposed an online method for a traffic algorithm in which every node (AR router in a facility) measures the one-way delay, traffic loss (packet loss) rate, and amount of traffic demand independently. A control service collects this network information; searches for a suboptimal solution, which is described by a combination of paths; and advertises new routing tables. The service starts to calculate the evaluation value of the combinations as a trigger to the scheduled traffic demand and the detection of network disorder. The service announces a new routing table for a suboptimal solution that reduces the packet losses above a certain threshold.

Evaluation examinations show that the method can evaluate combinations with sufficient accuracy and can obtain an optimal solution at a sufficient speed for an eight-node network using 160 thread computations. An evaluation examination on an eleven-node

network shows that the method works around network disorder, such as the emergence of large traffic amounts by searching for a suboptimal solution within several seconds. As future work, we intend to solve the problems regarding the amount of calculations based on the growth in the number of nodes used in an overlay network.

## 5.2 Future Works

In the decentralized stochastic approach, the reduction for size of network routing table and more realistic simulation evaluation is our future works. We are planning to implement self-assembly network which can autonomously partition the network routing domain.

And the implementation of the decentralized stochastic approach, quality of connection can be lost. It is necessary to improve the algorithm with more network information such as available bandwidth and network availability so as to afford both of high ability of packet processing and high quality of network communication.

Finally, in the centralized deterministic approach, it is strongly recommended to solve problems about amount of calculation according to growth of number of nodes consisted the overlay network.

OpenFlow implementation on the real network and evaluation with specific network applications will be done. In the future, these algorithm will be published as IETF RFC documents.

# Bibliography

- [1] 総務省総合通信基盤局電気通信事業部データ通信課. 我が国のインターネットにおけるトラフィック総量の把握, September 2013.  
URL [http://www.soumu.go.jp/menu\\_news/s-news/01kiban04\\_02000061.html](http://www.soumu.go.jp/menu_news/s-news/01kiban04_02000061.html).
- [2] Christian Poellabauer Waltenegus Dargie. *Fundamentals of Wireless Sensor Networks: Theory and Practice*. Wiley, 2010. ISBN 978-0-470-99765-9.
- [3] The Economist. Data, data everywhere, 2010.
- [4] Priscila Lpez Martin Hilbert. The world ’ s technological capacity to store, communicate, and compute information. *Science*, 332(6025):60–65, April 2011.
- [5] Adam Jacobs. The pathologies of big data. *Queue*, 7(6):10:10–10:19, July 2009. ISSN 1542-7730. doi: 10.1145/1563821.1563874.  
URL <http://doi.acm.org/10.1145/1563821.1563874>.
- [6] Jimmy Guterman. *Release 2.0: Issue 11 Big Data*. O’Reilly Media, Inc., 2009.
- [7] Ulf-Dietrich Reips Chris Snijders, Uwe Matzat. “big data”: Big gaps of knowledge in the field of internet science. *International Journal of Internet Science*, 7(1):1–5, 2012. ISSN 1662-5544.
- [8] Doug Laney. Application delivery strategies, 2001.  
URL <http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf>.
- [9] Gartner says solving ’big data’ challenge involves more than just managing volumes of data, 2011. URL <http://www.gartner.com/newsroom/id/1731916>.
- [10] 総務省. 平成 24 年度版 情報通信白書, July 2012.
- [11] Martin Fransman. *Telecoms in the Internet Age: From Boom to Bust To?* Oxford University Press, 2002. ISBN 0199257000.

- [12] S.F. Bush. A simple metric for ad hoc network adaptation. *Selected Areas in Communications, IEEE Journal on*, 23(12):2272–2287, Dec 2005. ISSN 0733-8716. doi: 10.1109/JSAC.2005.857204.
- [13] Stephen F. Bush. Active virtual network management prediction: Complexity as a framework for prediction, optimization, and assurance. In *Proceedings of the 2002 DARPA Active Networks Conference and Exposition*, DANCE '02, pages 534–, Washington, DC, USA, 2002. IEEE Computer Society. ISBN 0-7695-1564-9. URL <http://dl.acm.org/citation.cfm?id=874028.874138>.
- [14] Stephen F. Bush. Toward in vivo nanoscale communication networks: Utilizing an active network architecture. *Front. Comput. Sci China*, 5(3):316–326, September 2011. ISSN 1673-7350. doi: 10.1007/s11704-011-0116-9. URL <http://dx.doi.org/10.1007/s11704-011-0116-9>.
- [15] Jaidev P. Patwardhan, Chris Dwyer, Alvin R. Lebeck, and Daniel J. Sorin. Nana: A nano-scale active network architecture. *J. Emerg. Technol. Comput. Syst.*, 2(1): 1–30, January 2006. ISSN 1550-4832. doi: 10.1145/1126257.1126258. URL <http://doi.acm.org/10.1145/1126257.1126258>.
- [16] Alberto Medina, Anukool Lakhina, Ibrahim Matta, and John Byers. Brite: An approach to universal topology generation. In *Proceedings of the Ninth International Symposium in Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, MASCOTS '01, pages 346–, Washington, DC, USA, 2001. IEEE Computer Society. URL <http://dl.acm.org/citation.cfm?id=882459.882563>.
- [17] 小原 泰弘, 今泉 英明, 加藤 朗, 中村 修, and 村井 純. 広範なトラフィック要求に対応する負荷分散経路計算アルゴリズム. *情報処理学会論文誌*, 48(4):1627–1640, April 2007.
- [18] E. Rosen, A. Viswanathan, and R. Callon. Multiprotocol label switching architecture. *Network Working Group RFC 3031*, January 2001. <http://tools.ietf.org/html/rfc3031>.
- [19] 熊木 健二, 中川 郁夫, 永見 健一, 長谷川 輝之, and 阿野 茂浩. キャリアネットワークにおける MPLS TE LSP 確立に関するロードバランス手法の提案と評価. *情報処理学会論文誌*, 48(4):1616–1626, April 2007.
- [20] Zheng Wang. *Internet QoS: Architectures and Mechanisms for Quality of Service*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition, 2001. ISBN 1558606084.
- [21] 山田 仁, 高島 研也, 仲道 耕二, 宗宮 利夫, and 中後 明. トラヒックエンジニアリングシステムの実機評価. *電子情報通信学会技術研究報告. NS, ネットワークシステム*,

- 101(8):45–50, April 2001. ISSN 09135685.  
URL <http://ci.nii.ac.jp/naid/110003181136/>.
- [22] 中川 郁夫, 江崎 浩, 菊池 豊, and 永見 健一. MPLS を用いた広域分散 IX の実現. 情報処理学会論文誌, 43(11):3519–3529, November 2002. ISSN 03875806.  
URL <http://ci.nii.ac.jp/naid/110002711449/>.
- [23] 菊池 豊, 石原 丈士, 永見 健一, 楠田 友彦, 菱岡 裕男, 西内一馬, 羽田 友和, 水村 雅明, 正岡 元, 池田 浩志, 中川郁夫, and 江崎 浩. 異機種ルータの相互接続試験活動 : 新しいネットワークアーキテクチャの導入を促進するために. 電子情報通信学会技術研究報告. SS, ソフトウェアサイエンス, 106(15):19–24, April 2006. ISSN 09135685.  
URL <http://ci.nii.ac.jp/naid/110004718941/>.
- [24] E.J. Anderson and T.E. Anderson. On the stability of adaptive routing in the presence of congestion control. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 2, pages 948–958 vol.2, 2003. doi: 10.1109/INFCOM.2003.1208932.
- [25] Srikanth Kandula, Dina Katabi, Shantanu Sinha, and Arthur Berger. Dynamic load balancing without packet reordering. *SIGCOMM Comput. Commun. Rev.*, 37(2): 51–62, March 2007. ISSN 0146-4833. doi: 10.1145/1232919.1232925.  
URL <http://doi.acm.org/10.1145/1232919.1232925>.
- [26] 岸田 崇志, 前田 香織, and 河野 英太郎. ネットワーク障害物を乗り越えるテレビ会議用ゲートウェイの開発. 情報処理学会論文誌, 48(4):1552–1561, April 2007.
- [27] Minho JO, Hyoung Do KIM, and Hyogon KIM. An adaptive routing method for voip gateways based on packet delay information. *IEICE transactions on communications*, 88(2):766–769, February 2005. ISSN 09168516.  
URL <http://ci.nii.ac.jp/naid/110003222703/>.
- [28] H. Kashiwazaki and N.K. Takai. The simulation evaluation for an adaptive network routing by using latency information. In *Applications and the Internet (SAINT), 2011 IEEE/IPSJ 11th International Symposium on*, pages 480–485, 2011. doi: 10.1109/SAINT.2011.89.
- [29] 内田 真人, 亀井 聡, and 川原 亮一. オーバーレイネットワークによる qos ルーティング制御に関する評価. 電子情報通信学会技術研究報告. IN, 情報ネットワーク, 105(178):13–18, July 2005. ISSN 09135685. URL <http://ci.nii.ac.jp/naid/110003224889/>.
- [30] Lili Qiu, Yang Richard Yang, Yin Zhang, and Scott Shenker. On selfish routing in internet-like environments. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM

- '03, pages 151–162, New York, NY, USA, 2003. ACM. ISBN 1-58113-735-4. doi: 10.1145/863955.863974. URL <http://doi.acm.org/10.1145/863955.863974>.
- [31] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao. Overview and principles of internet traffic engineering. *Network Working Group RFC 3272*, May 2002. <http://tools.ietf.org/html/rfc3272>.
- [32] S. Vutukury and J.J. Garcia-Luna-Aceves. A distributed algorithm for multipath computation. In *Global Telecommunications Conference, 1999. GLOBECOM '99*, volume 3, pages 1689–1693 vol.3, 1999. doi: 10.1109/GLOCOM.1999.832451.
- [33] H. Michiel and K. Laevens. Teletraffic engineering in a broad-band era. *Proceedings of the IEEE*, 85(12):2007–2033, December 1997. ISSN 0018-9219. doi: 10.1109/5.650182.
- [34] P.E. Wirth. The role of teletraffic modeling in the new communications paradigms. *Communications Magazine, IEEE*, 35(8):86–92, August 1997. ISSN 0163-6804. doi: 10.1109/35.606035.
- [35] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow. RSVP-TE: Extensions to RSVP for LSP Tunnels. RFC 3209 (Proposed Standard), December 2001. URL <http://www.ietf.org/rfc/rfc3209.txt>. Updated by RFCs 3936, 4420, 4874, 5151, 5420, 5711, 6780, 6790.
- [36] Gianni Di Caro and Marco Dorigo. An adaptive multi-agent routing algorithm inspired by ants behavior. In *Proc. PART98 5th Annual Australasian Conference on Parallel and Real-Time Systems*, pages 178–185, 1998.
- [37] Marco Dorigo and Thomas Stützle. *Ant Colony Optimization*. Bradford Company, Scituate, MA, USA, 2004. ISBN 0262042193.
- [38] Marco Dorigo, Gianni Di Caro, and Luca M. Gambardella. Ant algorithms for discrete optimization. *Artif. Life*, 5(2):137–172, April 1999. ISSN 1064-5462. doi: 10.1162/106454699568728. URL <http://dx.doi.org/10.1162/106454699568728>.
- [39] G. Di Caro and M. Dorigo. Mobile agents for adaptive routing. In *System Sciences, 1998., Proceedings of the Thirty-First Hawaii International Conference on*, volume 7, pages 74–83 vol.7, January 1998. doi: 10.1109/HICSS.1998.649179.
- [40] Eric Bonabeau, Florian Henaux, Sylvain Guérin, Dominique Snyers, Pascale Kuntz, and Guy Theraulaz. Routing in telecommunications networks with “smart” ant-like agents. Working Papers 98-01-003, Santa Fe Institute, January 1998. URL <http://ideas.repec.org/p/wop/safiwp/98-01-003.html>.

- [41] 山口 直彦, 棟朝 雅晴, 赤間 清, and 佐藤 義治. リンク負荷メトリックに基づく遺伝的アルゴリズムによる負荷分散ルーティング. 情報処理学会論文誌, 43(7):2359–2367, July 2002.
- [42] 棟朝 雅晴, 高井 昌彰, and 佐藤 義治. 遺伝的アルゴリズムによる負荷分散機構を有する適応型ルーティング. 情報処理学会論文誌, 39(2):219–227, February 1998.
- [43] L. Barolli, A. Koyama, S. Motegi, and S. Yokoyama. An intelligent policing-routing mechanism based on fuzzy logic and genetic algorithms. In *Parallel and Distributed Systems, 1998. Proceedings. 1998 International Conference on*, pages 390 –397, December 1998. doi: 10.1109/ICPADS.1998.741104.
- [44] Barolli Leonard, Koyama Akio, Yamada Takako, and Yokoyama Shoichi. An integrated cac and routing strategy for high-speed large-scale networks using cooperative agents. 情報処理学会論文誌, 42(2):222–233, February 2001.
- [45] Ivan Stojmenovic and Xu Lin. Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks. *IEEE Trans. Parallel Distrib. Syst.*, 12(10):1023–1032, October 2001. ISSN 1045-9219. doi: 10.1109/71.963415. URL <http://dx.doi.org/10.1109/71.963415>.
- [46] Masato Tsuru, Tetsuya Takine, and Yuji Oie. Estimation of clock offset from one-way delay measurement on asymmetric paths. In *Proceedings of the 2002 Symposium on Applications and the Internet (SAINT) Workshops, SAINT-W '02*, pages 126–, Washington, DC, USA, 2002. IEEE Computer Society. ISBN 0-7695-1450-2. URL <http://dl.acm.org/citation.cfm?id=580055.829297>.
- [47] Albert-Lszl Barabasi and Rka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, October 1999. doi: 10.1126/science.286.5439.509. URL <http://dx.doi.org/10.1126/science.286.5439.509>.
- [48] Agilent Technologies. The journal of internet test methodologies., 2007.
- [49] 三屋 光史朗, 長 健二郎, 加藤 朗, and 村井 純. パケットサイズ分布からみた ip トラフィックの傾向. In *インターネットコンファレンス 2000 論文集, インターネットコンファレンス 2000*, pages 47–56, 2000. URL [http://www.internetconference.org/ic2000/papers/S02\\_03.pdf](http://www.internetconference.org/ic2000/papers/S02_03.pdf).
- [50] 岩間 司, 金子 明弘, 町澤 朗彦, and 鳥山 裕史. 高速ネットワークを利用した高精度時刻比較. 電子情報通信学会論文誌. D, 情報・システム, 89(12):2553–2563, December 2006. ISSN 18804535. URL <http://ci.nii.ac.jp/naid/110007380392/>.
- [51] 山田 雄介. IP ネットワーク上の時刻同期手法. 電子情報通信学会技術研究報告. CS, 通信方式, 105(280):1–6, September 2005. ISSN 09135685. URL <http://ci.nii.ac.jp/naid/110003283586/>.

- [52] 町澤 朗彦, 鳥山 裕史, and 岩間 司. パケット到着間隔 (pai) に基づいた時刻同期方式 (nw 性能管理, nw 品質, 一般). 電子情報通信学会技術研究報告. *CQ*, コミュニケーションクオリティ, 105(406):35–40, November 2005. ISSN 09135685.  
URL <http://ci.nii.ac.jp/naid/110004020820/>.
- [53] 菊池 豊, 中川 郁夫, 樋地 正浩, 八代 一浩, and 林 英輔. ジャパンギガビットネットワーク: 4. 地域間相互接続実験プロジェクト. 情報処理, 43(11):1171–1177, November 2002.
- [54] 近堂 徹, 西村 浩二, 相原 玲二, 前田 香織, and 大塚 玉記. 高品質動画像伝送における fec の性能評価. 情報処理学会論文誌, 45(1):84–92, January 2004.
- [55] Tao Ye, H.T. Kaur, S. Kalyanaraman, and M. Yuksel. Large-scale network parameter configuration using an on-line simulation framework. *Networking, IEEE/ACM Transactions on*, 16(4):777–790, 2008. ISSN 1063-6692. doi: 10.1109/TNET.2008.2001729.
- [56] H. Tamura, T. Okubo, K. Kawahara, Y. Oie, and Y. Inoue. Implementation and experimental evaluation of on-line simulation server for ospf-te. In *Hybrid Intelligent Systems, 2007. HIS 2007. 7th International Conference on*, pages 259–264, 2007. doi: 10.1109/HIS.2007.42.
- [57] 柏崎 礼生 and 高井 昌彰. 遅延時間情報に基づく適応的ネットワークルーティング. 情報処理学会論文誌, 47(12):3308–3318, December 2006.
- [58] 柏崎 礼生, 小林 悟史, 河合 修吾, 大石 憲且, and 高井 昌彰. 片方向遅延を用いたネットワークトラフィックの適応的負荷分散手法. 情報処理学会論文誌, 49(3):1194–1203, March 2008.
- [59] 菊池 豊, 藤井 資子, 山本 正晃, 永見 健一, and 中川 郁夫. 遅延計測による日本のインターネットトポロジーの推定. 情報処理学会研究報告. *DSM*, [分散システム/インターネットネットワーク運用技術], 2007(72):103–108, July 2007. ISSN 09196072.  
URL <http://ci.nii.ac.jp/naid/110006379748/>.
- [60] Vinay J. Ribeiro, Rudolf H. Riedi, Richard G. Baraniuk, Jiri Navratil, and Les Cottrell. pathchirp: Efficient available bandwidth estimation for network paths. In *In Passive and Active Measurement Workshop*, 2003.
- [61] 浜 崇之, 藤田 範人, and 地引 昌弘. トラヒック短期変動の影響を考慮した利用可能帯域測定方式. 電子情報通信学会技術研究報告. *CQ*, コミュニケーションクオリティ, 107(134):67–72, July 2007. ISSN 09135685.  
URL <http://ci.nii.ac.jp/naid/110006382805/>.

- 
- [62] J. Moy. Ospf version 2. *Network Working Group RFC 2328*, April 1998.  
URL <http://tools.ietf.org/html/rfc2328>.
- [63] Teerawat Issariyakui and Ekram Hossain. *Introduction to Network Simulator NS2*. Springer, 1st edition, 2009. ISBN 978-0-387-71759-3.
- [64] Thaler D. and Hopps C. Multipath issues in unicast and multicast next-hop selection. *Network Working Group RFC 2991*, November 2000.  
URL <http://tools.ietf.org/html/rfc2991>.