PAPER

Optical Plug and Play Technique for Automatic GMPLS Network Construction

Rie HAYASHI[†a)], Kaori SHIMIZU[†*], *Members*, *and* Kohei SHIOMOTO[†], *Fellow*

SUMMARY We propose a mechanism called "optical plug and play" for constructing GMPLS networks automatically. It offers lower operation effort and fast network construction, and avoids misconfiguration. Optical plug and play architecture has its procedure, a link-up search mechanism for OXCs, network and node architectures to realize optical plug and play, and an LMP extension to exchange the information between nodes necessary for identifying adjacent nodes. We implement prototypes of both OXCs and routers that support the optical plug and play proposal. Simulations and experiments confirm its performance and feasibility.

key words: *plug and play, optical network, WDM, GMPLS*

1. Introduction

Wavelength division multiplexing (WDM) technology expands the capacity of optical fiber. WDM networks consist of routers and optical transport nodes such as ROADM (reconfigurable add-drop-multiplex or) and OXC (optical cross-connects), and WDM transport links. Neighbor nodes are interconnected by WDM transport links, each of which carries a bunch of wavelength links. Network topology has been changing from a relatively simple one such as mono-ring and multi-ring topology to something more complex such as mesh topology to flexibly adapt to changes in the traffic demand pattern and to reduce capital expenditure (CAPEX). On the other hand, operational expenditure (OPEX) will be high given the complexity of the topology. In order to reduce the OPEX, we need a sophisticated network management and operation system.

GMPLS (Generalized Multi Protocol Label Switching) control plane (c-plane), which has been developed and standardized at the IETF [1], is expected to reduce the OPEX because of its autonomous distributed network controllability, which leads to operation labor reduction. GMPLS c-plane has a routing protocol, which can be used as a topology discovery mechanism. All nodes advertise link-state in order to collect full topology information. Once a WDM link is installed between two nodes and those two nodes recognize that the WDM link is up, they start to advertise the link-state associated with the WDM link to all other nodes in the network by using the flooding mechanism implemented in GMPLS routing protocol [2]. Currently however, we need to manually configure interface addresses of wavelength links before we advertise link-state using a GMPLS routing protocol for an automatic topology discovery mechanism. Given the number of optical fibers between neighbor nodes, each of which again carries a bunch of wavelength links (say 40, 80, 160 wavelength links per fiber), it is a tedious task to configure the interface addresses of wavelength links between neighbor nodes. Additionally, configuration mistakes are possible because of the large number of wavelength links to be configured. Moreover, every time we change the physical topology of the WDM network (e.g., rewiring, removal, addition of optical fibers between neighbor nodes), we need to configure interface addresses of wavelength links between neighbor nodes.

One of the GMPLS protocols is LMP (link management protocol) [7]. LMP roles include control channel management and TE-link property correlation, and its options include link connectivity verification and fault management of data links. During these management tasks, parameters such as node ID, interface addresses, and TE-link addresses are exchanged between adjacent nodes. Each node, therefore, gets adjacent node information by LMP. However, to prepare for LMP operation, manual tasks include deciding the interface and TE-link addresses, configuration setting of each node, and understanding the existence of and the number of adjacent nodes. This means that it is impossible to start LMP automatically without a considerable amount of parameter configuration.

This paper proposes a solution that suppresses the tasks of configuring addresses of wavelength links between neighbor nodes. The contributions of this paper are establishing an optical plug and play (PnP) architecture for WDM networks by combining the following innovative elements and confirming its performance and feasibility. Optical PnP includes the automatic assignment of interface addresses, parameter information exchange between neighbor nodes, database construction, and so on. We propose node architectures and functions deployment to realize an optical PnP technique targeting for routers and OXCs [5]. We also propose a link-up search mechanism for OXCs which is passive and has no electrical mechanism [3], [4], [6]. A challenge is how to detect a newly connected link and identify identifier and interface addresses of neighbor nodes. Moreover, we propose a protocol extension for a fast optical PnP process. Information exchange between neighbor nodes is necessary for PnP automatic configuration, and we use an extended version of LMP [7], [8] for this information exchange. We implement a prototype that supports the functions necessary

Manuscript received September 13, 2010.
Manuscript revised January 26, 2011.

[†] The authors are with NTT Network Services Systems Laboratories, NTT Corporation, Musashino-shi, 180-8585 Japan.
[*] Presently, with NTT Communications Corporation.
a) E-mail: hayashi.rie@lab.ntt.co.jp

DOI: 10.1587/transcom.E94.B.1933

for optical PnP, and demonstrate it in a WDM network [9]. We evaluate its performance in terms of the time required to construct a network, and confirm its advantages and feasibility.

The organization of this paper is as follows. Section 2 introduces the optical PnP procedure. Section 3 describes network and node architectures to realize optical PnP and implementation methods. Section 4 presents a link-up search mechanism for OXCs. In Sect. 5, an LMP extension is described to apply optical PnP. Section 6 presents evaluation results of optical PnP performance by both simulation and experiment. Finally, Sect. 7 summarizes the key points.

2. Optical PnP Overview

2.1 Purposes and Merits

We establish an optical PnP technique in order to automatically set network parameters and to minimize OPEX (operation expenditure) incurred in providing services. By the mechanisms described in the following sections, optical PnP enables operation labor reduction, configuration mistake avoidance, and fast network construction.

2.2 Optical PnP Procedure

For optical PnP we propose automatic parameter assignment (technique-1) and exchange (technique-2) for configuring parameters such as interface address at a local node. Here, local node means the node we are focusing on, while its counterpart is called the remote node. These techniques are over viewed in Fig. 1.

2.2.1 Automatic Parameter Assignment

Technique-1 enables GMPLS parameters of a local node to be set when triggered by link-up (optical cable connection). Upon detecting link-up (detecting mechanism is described in detail later), the node sends a notification to a controller dedicated to the node. At this time, the controller generates a set of local node parameters consisting of interface address, TE-link [7] address, and so on. In order to make these parameters unique in the network, each node prepares a list of assignable parameters. When address assignment is requested, a node selects one of the addresses in the list and assigns it to the target interface or TE-link.

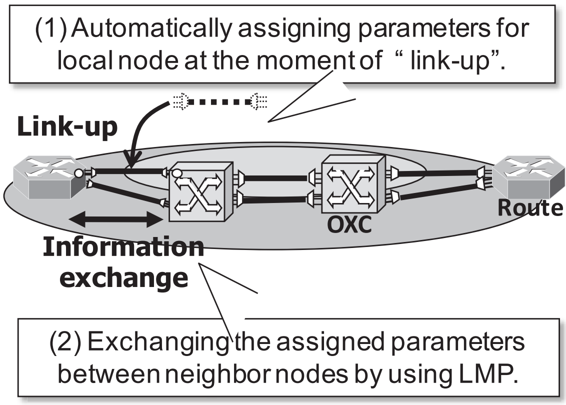
(1) Automatically assigning parameters for local node at the moment of “ link-up”.
Link-up
OXC
Router
Information exchange
(2) Exchanging the assigned parameters between neighbor nodes by using LMP.

Fig. 1 Key techniques for optical PnP.

2.2.2 Automatic Parameter Exchange

Technique-2 enables GMPLS parameters of a remote node to be set by exchanging messages between neighbor nodes. We use LMP for exchanging messages, which include configuration messages used on c-plane channels and a test message used on d-plane channels. The messages include local node parameters generated by technique-1. A node receiving the messages stores them as remote node's parameters. After that, the node advertises the local and remote parameters through c-plane to all network nodes.

2.3 Optical PnP Procedure

The main difference between routers and OXCs with regard to optical PnP, other than the key techniques explained above, is whether a search mechanism to find link-up interfaces is necessary or not. In order to detect link-up at OXC, we propose an OXC architecture and a link-up search mechanism in Sects. 3 and 4, respectively. Figure 2 explains the optical PnP procedure.

Step 1. OXCs operate the link-up search mechanism, while routers always prepare for cable connection by sending signals.
Step 2. For OXC-OXC connection, cable-connected interfaces are detected by a link-up search mechanism. For the router-OXC case, when continuity test signals are sent to a router through an OXC interface selected by the search mechanism, both nodes detect the cable-connected interface.
Step 3. An address is assigned to the target interface in each node.
Step 4. The assigned address information is exchanged between neighbor nodes by LMP. By recognizing parameters related to local and remote nodes, each node automatically creates database entries.
Step 5. Finally, nodes are ready for operation. For example, nodes start OSPF-TE advertisement to advertise their

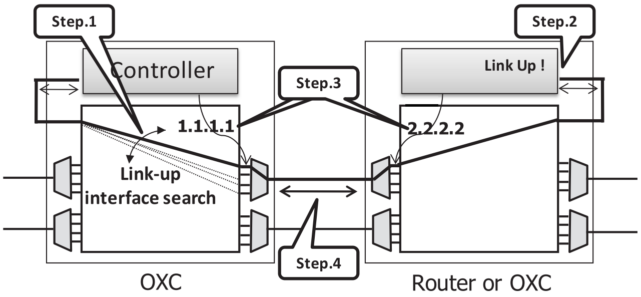

Fig. 2 Optical PnP procedure.

routing information collected by LMP.

LMP is a protocol to exchange node information between adjacent nodes. PnP, therefore, first operates LMP and adjacent nodes exchange information. Based on the information collected by LMP, each node starts OSPF-TE advertisement, by which each node advertises and collects routing information of all nodes in its domain.

Note that the proposed optical PnP technique is assumed to be applied to one domain covered by OSPF-TE because we assume that all routing information is advertised by OSPF-TE.

3. Architecture and Function Deployment

In this section, we describe a network architecture based on optical PnP as well as a node architecture, and function deployment in the node. We target OXCs and routers as the optical PnP nodes. In particular, we assume that the OXC has only cross-connect function and no electrical function (i.e., 2R/3R, λ conversion). This means that the OXC does not detect link-up by itself.

3.1 Network Architecture

Figure 3 shows the network architecture that implements optical PnP. We assume that nodes such as OXCs and routers are connected by WDM links. In optical PnP, at least one wavelength in a WDM link is dedicated to an in-band control channel to exchange the information necessary for PnP between neighbor nodes. Although an out-of band control channel is possible instead of in-band one, the PnP link-up search mechanism, which is described in Sect. 4, for the out-of-band control channel is out of the scope in this paper. All wavelengths other than the control one support data channels.

Neighbor nodes use LMP to exchange assigned interface addresses between themselves.

3.2 OXC Architecture

Figure 4 shows the proposed OXC architecture for optical PnP. It consists of an $N \times N$ optical switch, such as a MEMS or a thermo-optical switch, and a controller including an electric processing unit and a control element. The electric processing unit is attached to a pair of input and output interfaces of the optical switch. It implements a link-up function to identify cable (ex. fiber) connection, and the GMPLS protocol transfer function that exchanges messages with neighbor nodes. Many functions are assigned to each optical switch interface. D-plane channel interfaces switch data signals to appropriate neighbor nodes. C-plane channel interfaces support the in-band control channel and send/receive control messages sent from its own/neighbor node's electric processing unit. A link-up interface transfers continuity test signals to identify link-up. It is connected to a corresponding interface of the electric processing unit and is cross-connected to a d-plane or c-plane channel interface. Similarly, the interface used for sending/receiving GMPLS protocols is cross-connected to one of the c-plane channel interfaces. The parameters necessary to decide the specification of the OXC architecture are detailed below.

- ➢ Switch size: N interfaces in the optical switch are constituted by c-plane and d-plane channel interfaces, link-up interface (P_{linkup}), and GMPLS protocol exchange interfaces (P_{prot}).
- ➢ Number of wavelengths multiplexed in a WDM link: The number of wavelengths multiplexed in a WDM link is the sum of wavelengths for d-plane (λ_{data}) and c-plane (λ_{cc}). The number of WDM links connected to the optical switch equals the maximum number of neighbor nodes the optical switch can connect to (C_{max}).
- ➢ Wavelength: Range (ex. C-band, L-band) and interval (ex. DWDM, CWDM) should be decided.
- ➢ Maximum transmission load: Estimated as the sum of signal transmission speeds of all wavelengths assigned to the d-plane.
- ➢ Number of interfaces necessary for electric processing unit: Protocol transfer function in electric processing unit needs the same number of interfaces (T_{prot}) as

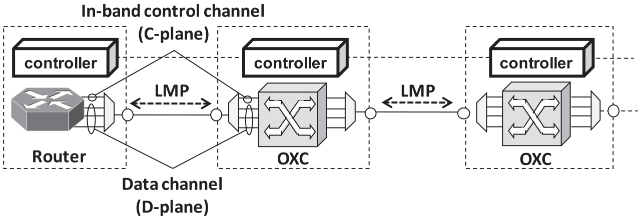

Fig. 3 Optical PnP network architecture.

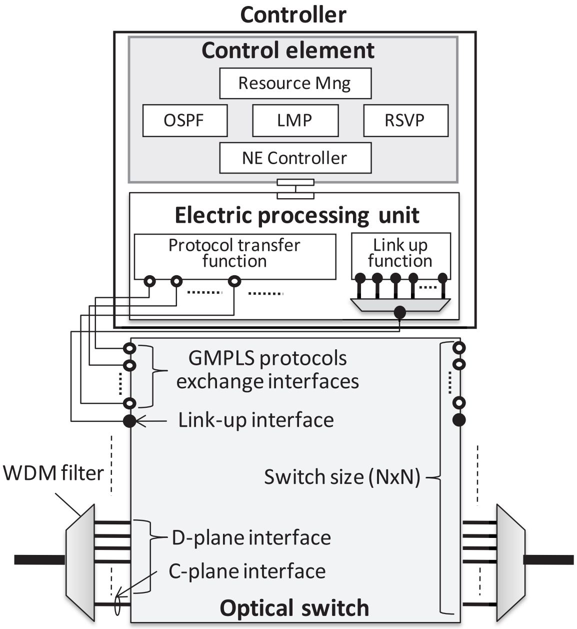

Fig. 4 Proposed OXC architecture.

neighbor nodes to exchange protocol messages. Similarly, link-up function in electric processing unit needs to send/receive all kinds of wavelength signals the optical switch supports to detect the link-up of arbitrary interfaces, and needs the same number of interfaces (T_{linkup}) as wavelengths supported in the optical switch. These interfaces (T_{linkup}) are multiplexed and the multiplexed signals are sent to/from the optical switch.

These parameters have following relation;

$$N \geq (\lambda_{cc} + \lambda_{data}) \times C_{max} + P_{port} + P_{linkup} \tag{1}$$

$$P_{port} = T_{port} = C_{max} \tag{2}$$

$$T_{linkup} = \lambda_{cc} + \lambda_{data} \tag{3}$$

$$P_{Vlinkup} = 1 \tag{4}$$

$$\lambda_{cc} = 1 \tag{5}$$

Controller is also necessary to control PnP mechanism based on the proposed switch architecture (Fig. 4). The controller includes a network element (NE) controller to control the optical switch, such as cross-connection of switch interfaces for transferring test signals for link-up detection and data. LMP function exchanges information between adjacent nodes during the PnP process and collects the information needed for automatic configuration. Resource manager holds a database. It holds information of both the node itself and adjacent nodes. OSPF-TE and RSVP-TE [10] functions are utilized to set up optical paths after the PnP process. Former is for advertising and collecting network information, and the latter is for path set-up signaling. GMPLS-Engine [11] was used to realize GMPLS functions such as LMP, OSPF-TE, and RSVP-TE in the controller.

3.3 Router Architecture

We used a commercially-available GMPLS router to implement a prototype that supported the proposed PnP mechanism by using SDK (software development kit) [12]. Figure 5 shows the function blocks as implemented in a router.

Three daemons dedicated to PnP were installed in commercial GMPLS routers. “Event detection” daemon detects the link-up event when a cable is connected to a router interface. After the daemon notifies the controller of the cable-connection event, resource manager in the controller assigns an address to the notified interface. “Configuration operation” daemon creates the router configuration necessary for basic router operation after assigning addresses to link-up interfaces and exchanging information with adjacent nodes. “Send/receive packet” daemon transfers GMPLS protocol messages (as packets) through in-band control channels. For example, LMP Config/TestStatusSuccess/LinkSummary messages are exchanged during the PnP procedure, OSPF-TE messages including link-up interface information are advertised, and RSVP-TE messages are sent to set-up optical paths.

The router controller is similar to the one for OXC. It implements GMPLS protocol functions, resource manager, and NE controller. GMPLS protocol functions include LMP, OSPF-TE and RSVP-TE. However, we inactivated OSPF-TE and RSVP-TE in the router controller, and instead utilized the router’s protocol functions.

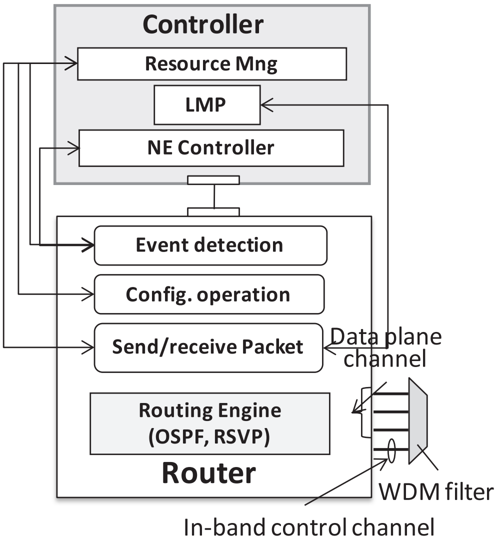

Fig. 5 Proposed router architecture.

4. Link-Up Search Mechanism

4.1 Random Search Mechanism

This paper assumes that the OXC is passive and has no electrical mechanism. Some mechanisms, therefore, are necessary to detect link-up. This section explains how OXCs detect the link-up of arbitrary interfaces and techniques for fast link-up search.

In order to detect an interface connected to an adjacent node, the controller of each OXC sends continuity test signals. In the continuity test signals exchange procedure, if (1) a controller interface at one of nodes (ex., node A) is interconnected through the interface connected to the remote node (ex., node B) and if (2) the controller interface at the other node (~ node B) is interconnected through the interface connected to the remote node (~ node A), the controllers at both sides successfully receive the test signals. Otherwise the test signal exchange procedure fails.

We should note that both sending and receiving nodes have to operate independently to achieve the neighbor discovery functionality. The question here is how to decide which remote/local interface of the d-plane link has to be interconnected to the controller at both nodes without communicating with each other.

In order to resolve the issue, we propose a method called the “random search” mechanism. The random search mechanism allows remote and local nodes to operate independently. The idea is that each node randomly selects an interface for the d-plane link to connect to the controller. Each node tries another interface after it tests one of the interfaces. Each node repeats this procedure until all interfaces have had their address decided.

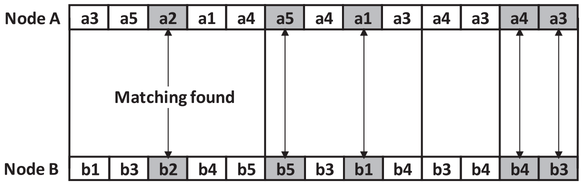
Node A
a3 a5 a2 a1 a4 a5 a4 a1 a3 a4 a3 a4 a3
Matching found
Node B
b1 b3 b2 b4 b5 b5 b3 b1 b4 b3 b4 b4 b3

Fig. 6 Example of random search sequence.

Figure 6 illustrates an example. We assume that the interface addresses of the wavelength link connecting nodes A and B are a_i and b_i ($1 \leqq i \leqq 5$) respectively. In the first cycle, node A decides to test the interfaces in the order of a_3, a_5, a_2, a_1, and a_4 while node B does so in the order of b_1, b_3, b_2, b_4, and b_5. In this cycle we find the matching interfaces a_2 and b_2. In the second cycle, both nodes randomly and independently decide the order of the interfaces to be tested after they remove the already matched interfaces, here, they are a_2 and b_2. The same procedure is repeated until there is no interface remaining unmatched. Even though we assumed that the cycles of both nodes are synchronized in Fig. 6 for the sake of brevity, they do not need to be synchronized.

4.2 Speed-Up of Link-Up Search Mechanism

One weakness of the random search mechanism described above is that it takes longer to detect link-up as the number of pairs of search target interfaces increases because of combinatorial explosion. We, therefore, propose a solution that considers WDM groups. Here, wavelengths accommodated in the same WDM filter (ex. Arrayed-Waveguide Grating: AWG) are regarded as occupying the same WDM group. The proposed solution takes advantage of the assumption that adjacent nodes are connected by a WDM link unit, meaning that the adjacent nodes are connected with a set of control and data channels belonging to the same WDM link at the same time. It has two key points;

(a) Control channel interfaces are the initial search target

In optical PnP, at least one wavelength in a WDM link is dedicated to an in-band control channel to exchange information necessary for PnP between adjacent nodes, and other wavelengths are used as data channels. We propose to search one of the control channel interfaces first followed by the corresponding (i.e., belonging to a same WDM group as the control channel) data channel interfaces. At first, as shown in Fig. 7, pairs of control channel interfaces at adjacent nodes are searched randomly. This random search for control channel interfaces is necessary in order to detect connection via arbitrary adjacent nodes at arbitrary timing. After one of control channel interfaces is successfully linked-up, a random search starts for data channel interfaces belonging to the same WDM group as the control channel interface. When link-up detection of both the control channel interface and data channel interfaces belonging to the WDM group finishes, the same procedure is repeated until there is no remaining interface whose address is undecided. Restricting the number of random search targets improves matching probability and link-up search is completed more rapidly. Note that this mechanism requires that interfaces used for control channels be established in advance.

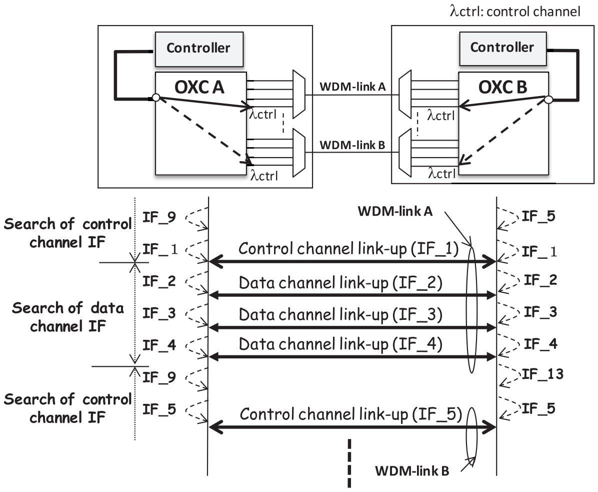
λctrl: control channel
Controller
OXC A
λctrl
λctrl
WDM-link A
WDM-link B
Controller
OXC B
λctrl
λctrl
Search of control channel IF
Search of data channel IF
Search of control channel IF
IF_9
IF_1
IF_2
IF_3
IF_4
IF_9
IF_5
WDM-link A
Control channel link-up (IF_1)
Data channel link-up (IF_2)
Data channel link-up (IF_3)
Data channel link-up (IF_4)
Control channel link-up (IF_5)
WDM-link B
IF_5
IF_1
IF_2
IF_3
IF_4
IF_13
IF_5

Fig. 7 Fast link-up search mechanism.

(b) Set order of search target interfaces

When data channel interfaces belonging to a certain WDM group are searched randomly after procedure (a), it still takes too long to check all interface combinations, because each WDM group has many wavelengths. Our solution is to set the order in data channel interface link-up search as shown in Fig. 7 (ex. increasing order of interface identifier). As a result, the link-up search time for one interface pair becomes one slot length, and the time necessary for searching all data channel interfaces in a WDM group is reduced to "unit slot length × the number of data channel interfaces". Here, slot represents the search time of one of the search target interfaces.

4.3 Link-Up Search Mechanism Performance

A numerical evaluation was made to confirm the performance of our proposed search mechanism.

First, we established the stochastic analytical model shown in the Appendix to confirm the random search mechanism. Figure 8 shows elapsed slot and probability distribution computed based on the model. We observe from Fig. 8 that the more search target interfaces (N) there are, the wider each probability distribution becomes. Additionally, the probability against elapsed time has its peak at the left side of the distribution.

Figure 9 shows simulation results of the expectation time required to complete link-up search with the probability of 97.5% versus the number of link-up target interfaces. Our proposed method (a) (control channel interfaces are the initial search target) shortens link-up search time by up to 44%. Moreover, by setting the search interface order with proposal (b) (set order of search target interfaces), the search time is additionally decreased by 40%. As a result, our proposed link-up search mechanisms keep the search time prac-

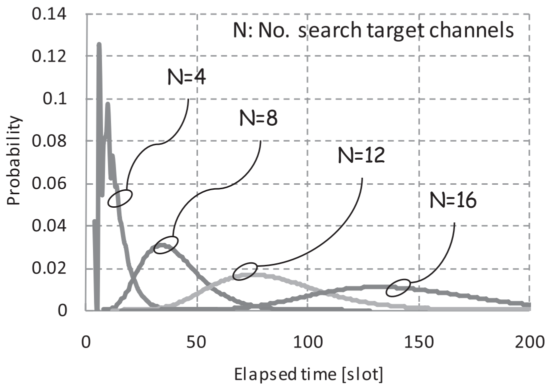

Fig. 8 Elapsed slot and probability distribution.

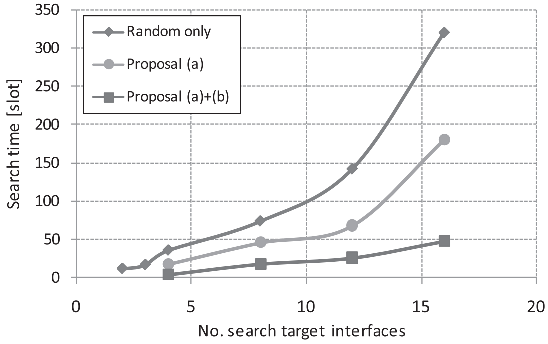

Fig. 9 Simulation results of expectation time required to complete link-up search (No. of multiplexed is 4).

tical even for a large number of search targets (i.e., interfaces), and thus ensure rapid completion of the PnP process as well as link-up search.

5. LMP Extension

GMPLS parameters of a remote node are set automatically by exchanging information between neighboring nodes. We extend LMP to exchange this information. There are two main extensions. One is to add LOCAL NODE ID object to a Test message. The other is that BeginVerify/EndVerify messages are not used in the link verification process.

5.1 LMP Message Extension

Our LMP extension proposal is based on the link verification process defined in [7]. First, we explain the standard LMP link verification process. The process starts with BeginVerify and BeginVerifyAck exchange between neighbors. A Test message is sent from one node through a data channel to check continuity. If the neighbor node receives the Test message, it replies with a TestStatusSucess message. Test and TestStatusSuccess message exchange is repeated until all data channels are checked. Finally, EndVerify message is sent to indicate process completion. VERIFY_ID object in BeginVerifyAck message is used to uniquely identify the Verification process from multiple LMP neighbors and/or parallel Test procedures. The VERIFY_ID object contains a node-unique value that is assigned by the generator of the BeginVerifyAck message. A node identifies an interface and a node that has sent a Test message by checking both VRIFY ID and LOCAL INTERFACE ID included in the Test message.

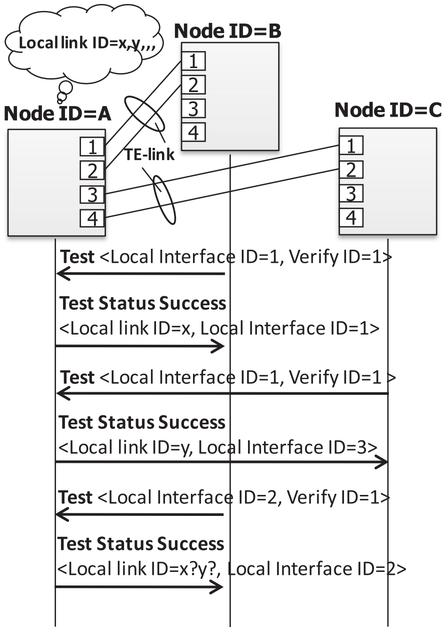
Local link ID=x,y,,,
Node ID=B
1
2
3
4
Node ID=A
1
2
3
4
TE-link
Node ID=C
1
2
3
4
Test <Local Interface ID=1, Verify ID=1>
Test Status Success
<Local link ID=x, Local Interface ID=1>
Test <Local Interface ID=1, Verify ID=1 >
Test Status Success
<Local link ID=y, Local Interface ID=3>
Test <Local Interface ID=2, Verify ID=1>
Test Status Success
<Local link ID=x?y?, Local Interface ID=2>

Fig. 10 Problem in identifying one PnP neighbor among multiple LMP neighbor nodes.

The following problems occur when a node tries to identify a PnP neighbor from among multiple LMP neighbor nodes (Fig. 10). The assumption is that node A exchanges LMP between nodes B and C for link construction by PnP. Node A receives Test messages from both node B and C, and both Test messages have the same description (LOCAL INTERFACE ID=1, VERIFY ID=1). Upon receiving a Test message, Node A needs to reply with a TestStatusSuccess message including LOCAL LINK ID as TE-link information. Upon receiving two Test messages, Node A is able to understand that these two Test messages come from different nodes since they include the same INTERFACE ID =1, and replies with TestStatusSuccess messages with different LOCAL LINK ID to each node.

A problem occurs when Node A receives two Test messages with (LOCAL INTERFACE ID=1, VERIFY ID=1) and (LOCAL INTERFACE ID=2, VERIFY ID=1), i.e. same VERIFY ID but different LOCAL INTERFACE ID. It is impossible for Node A to identify which node sent which Test message. Node A, therefore, does not understand with who *m* the cable is connected, and is unable to assign LOCAL LINK ID for a TestStatusSuccess message.

This paper proposes to add LOCAL NODE ID to each Test message in order to identify the node that sent a Test message, or the node to which the cable was just connected. Here, the Test message that is extended is the one used for PnP. Extension does not apply to the Test message used during regular link verification for physical connectivity verification.

The other problem is as follows. For implementing the LMP procedure, the number of data channels has to be set in advance. When the link verification process, which is a continuity test for data channels, begins, a node sends a BeginVerify message to indicate it is going to start link verification. A node that receives the BeginVerify message replies with a BeginVerifyAck message, and the continuity test process begins and test messages are exchanged. When link verification of all data channels is complete, EndVerify message is sent. In a PnP network, the number of data channel interfaces to which cables are actually connected is not known. This means that setting the data channel interface in advance is not possible and the timing to finish link verification for data channels as a trigger of link-up is not available. We, therefore, propose an alternative to BeginVerify/BeginVerifyAck/EndVerify messages. When a node confirms the link-up of any data channel interface, it immediately sends a Test message to the neighbor node.

5.2 Extended LMP Procedure

Figure 11 illustrates the extended LMP procedure and the objects contained in each LMP message when links are identified by optical PnP.

First, a cable is set between two nodes. After control channel interface link-up, procedures for Control Channel Management follow. Messages are transmitted through the control channel. In the Control Channel Management procedure, a Config message is sent from one node to the other node which replies with the corresponding ConfigAck message. To avoid ambiguity, the message exchange policy preferentially selects the node with the higher node identifier. The node with the lower node identifier must stop transmitting its message and respond to the message it receives. After the Control Channel Management procedure, nodes have the information necessary to activate control channels such as CCID (control channel identifier) and node ID of both nodes.

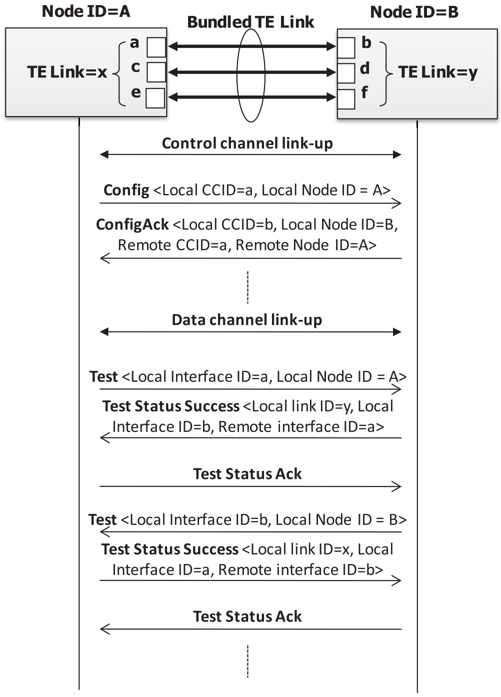
Node ID=A
Bundled TE Link
Node ID=B
TE Link=x
a
c
e
b
d
f
TE Link=y
Control channel link-up
Config <Local CCID=a, Local Node ID = A>
ConfigAck <Local CCID=b, Local Node ID=B, Remote CCID=a, Remote Node ID=A>
Data channel link-up
Test <Local Interface ID=a, Local Node ID = A>
Test Status Success <Local link ID=y, Local Interface ID=b, Remote interface ID=a>
Test Status Ack
Test <Local Interface ID=b, Local Node ID = B>
Test Status Success <Local link ID=x, Local Interface ID=a, Remote interface ID=b>
Test Status Ack

Fig. 11 Sequence of extended LMP.

Next, when data channel interface link-up is detected, the Link Verification procedure is initiated. LMP messages are transmitted through the corresponding control channel except for Test message, which is transmitted through a data channel. These are extended LMP messages and are required for optical PnP. At the beginning, a Test message is sent from one node to the other node without BeginVerify/BeginVerifyAck exchange. By adding NODE ID to the Test message, a node receiving the Test message which has INTERFACE ID as well as NODE ID, recognizes the correspondence between its interface, which was just linked to, and the adjacent node's interface. After receiving the Test message, corresponding TestStatusSuccess message is returned. Moreover, TestStatusAck message is used to acknowledge receipt of the TestStatusSuccess.

After the Link Verification procedure, nodes have the information necessary to activate data channels such as link ID and interface ID of both local and remote nodes. When link-up of other control channel interfaces or data channel interfaces is detected, the same procedures are used. After these procedures, node configuration is automatically made and network construction is automatically completed.

6. Experiments

6.1 Implementation

We constructed an optical network by PnP i.e., detecting link-up, assigning IF addresses and setting the configuration automatically. Figures 12 and 13 show PnP as implemented on commercially-available nodes, and implementation details of our PnP prototype, respectively.

The optical switch is thermo-optical and has 16×16 in-

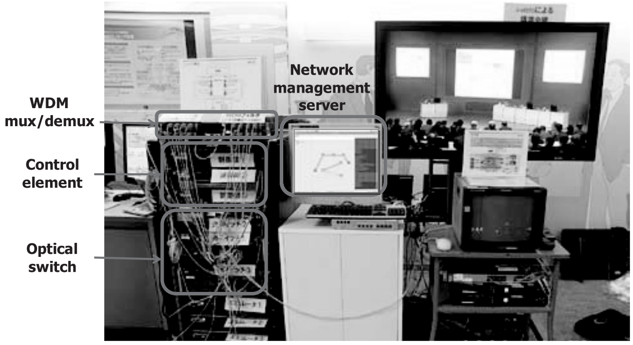

Fig. 12 Image of PnP implemented on commercially-available nodes. (Routers are located remotely.)

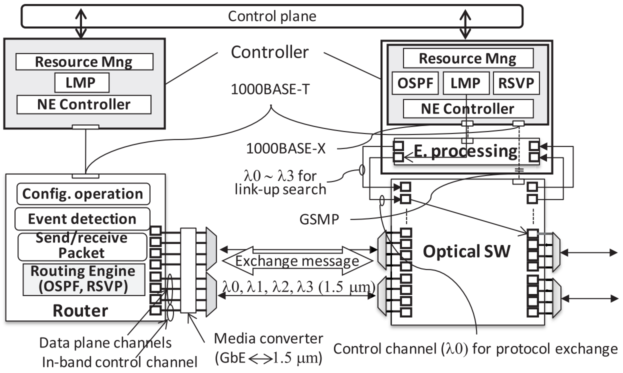

Fig. 13 Implementation details.

Table 1 Margin specifications.

	Targets	Specification
OXC	Optical switch	16x16 channels (12 for d-plane, 3 for protocol exchange, 1 for link-up), thermo-optical switch
		4-channel WDM filters
	Electric processing unit	Media converter (1.5mm↔1000BASE-X)
		4- channel WDM filters
	Controller	Dualcore intel® Xeon® processor (3.0GHz), 1000BASE-X and GbE IF, Cent-OS
		NE controller, LMP, OSPF-TE, RSVP-TE, Resource manager
	Protocol between optical switch and control element	GSMP through GbE IF
Router	Router	OSPF-TE, RSVP-TE
		GbE Ifs
		4-channel WDM filters
		Media converter (1.5mm↔1000BASE-X)
	Control element	Dualcore intel® Xeon® processor (3.0GHz), 1000BASE-X and GbE IF, Cent-OS
		NE controller, LMP, Resource manager

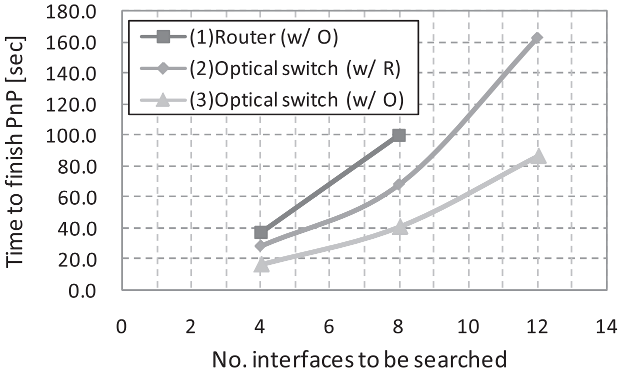

Fig. 14 Experimental results of PnP completion time.

terfaces. Three wavelength division and multiplex filters are attached to the optical switch for d-plane channels, while the remaining channels are connected to an electric processing unit and play the roles of either link-up interface or protocol exchange interface. The electric processing unit includes a media converter to convert 1.5 μm WDM optical signals to 1000BASE-X signals, which controller accepts, and a wavelength division and multiplex filter to multiplex several signals. The multiplexed signals are, as a result, able to jointly pass though a link-up interface (P_{linkup}) of the optical switch. Controller uses the general switch management protocol (GSMP) [13] to control the optical switch (ex. cross-connection).

A router has GbE interfaces. GbE signals sent from a router interfaces enter a media converter and are converted to 1.5 μm signals. The 1.5 μm signals output by the media converter are multiplexed by WDM filters; the WDM link connects a router and an optical switch. The controller is the same as an OXC except for the PnP functions supported.

In the optical network, each WDM link has four wavelengths, and there is one TE-link between neighbor nodes.

Table 1 shows specific implementation parameters of our PnP system.

For PnP, the following parameters are configured in advance; interfaces for control and data channels, node ID, WDM group to which control and data channels belong, GMPLS protocol exchange interfaces and link-up interface.

6.2 Optical PnP Performance

First, we evaluated the time needed to complete network construction by PnP. Here, a definition of PnP completion time in an OXC is from start of link-up search after connecting all of the cables to finishing LMP message exchange for the final linked interface. Similarly, a definition of PnP completion time in a router is from start of link-up search after connecting all of the cables to completion of router configuration registration for the final linked interface. We measured three PnP-neighbor pairs; measurement targets were a route and an OXC, and the PnP neighbor was OXC, router or OXC. Figure 14 shows that it took only 80 sec to complete PnP between OXCs with 12-PnP search target interface pairs. Additionally, it took only 160 sec to complete PnP in an OXC with 12-interface pairs to adjacent router, much faster than manual setting. A router takes longer than an OXC due to its use of PnP daemons; moreover, the modules in the controller are physically separated and must cooperate via SDK APIs. We believe that processing speed can be increased if GMPLS daemons (OSPF, RSVP, LMP, and PnP) cooperate with each other. Here, the link-up search mechanism consisted of proposals (a)+(b) described in Sect. 4.2.

Next, we analyzed the time necessary to set a control channel. The definition of control channel set up time in an OXC is from link-up detection to starting OSPE-TE hello message advertisement, while that of a router is from link-up detection to a completion of router configuration registration. The result was that it took 8~10 sec at both routers and OXCs. The slowest process in the OXC was to start sending an OSPF-TE hello message after finishing the LMP process. While, that in the router was to complete router configuration registration for the control channel. We believe that processing speed can be raised by having the GMPLS daemons in the routers cooperate with each other.

Finally, data channel set-up time was measured. The definition of data channel set up time in an OXC is from link-up detection to starting OSPE-TE TE-LSA advertisement, while that of a router is from link-up detection to com-

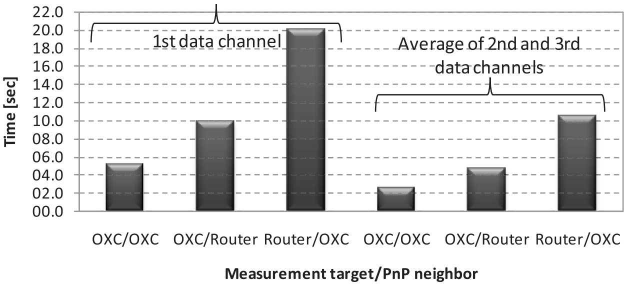

Fig. 15 Experimental results of data channel set-up time.

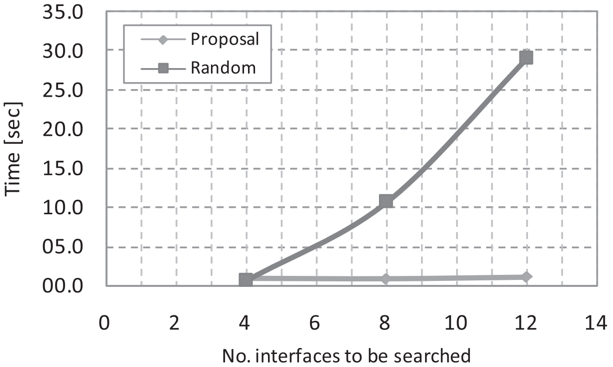

Fig. 16 Experimental results of required time to search first link-up interface between OXCs (No. of multiplexed is 4).

pletion of router configuration registration. Figure 15 shows that the first data channel to be linked-up first takes longer to set-up. This is because the 1st data channel interface has to find the corresponding control channel interface which is then followed by TE-link setting. In addition, data channels belonging to a router take longer to set-up because of the separation of the controller and PnP daemons as well as incomplete cooperation among daemons. The slowest process in the OXC was sending a LMP TestStatusSuccess message after receiving a Test message. That in the router was sending a LMP Test message after detecting link-up, which needs cooperation with an outside controller which was realized here by TCP.

These experiments confirm that PnP offers faster network construction than manual tasks.

6.3 Link-Up Search Performance

We conducted a numerical evaluation to confirm the performance of the search mechanism proposed in Sect. 4.3. Here, we assessed the time required to find the first link-up. In Fig. 16, our search mechanism finds the link-up quickly even if there are many search target interfaces. This is because, unlike random search, the proposed mechanism restricts the number of random search targets, which improves matching probability.

Figure 17 shows PnP completion time between OXCs with link-up search mechanisms. Each measurement point shows maximum, average, and minimum values, and curves run through the average values. Our search mechanism takes less than 100 sec to complete PnP (including link-up search, parameter exchange, and so on) between adjacent OXCs having 12-interface pairs to be searched, which is faster than is possible with the random search mechanism. When PnP neighbors are routers, the proposal and random search take the same time to finish. This is because routers always prepare for link-up and so do not need a search mechanism.

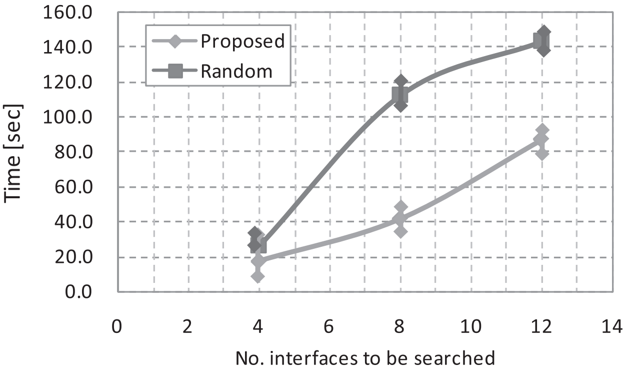

Fig. 17 Experimental results of required time to complete PnP at an OXC.

Finally, we compare the link-up search times as determined by numerical and experimental evaluations. Considering the above experimental results, set-up of control and data channels at a 16-interface OXC (divided into three WDM groups, each of which includes one control channel and three data channels, i.e. 12 PnP search target interfaces) requires the following time. Control channels require 9 sec on average × three control channels = 27 sec. Here, 9 sec comes from our experimental result described in Sect. 6.2. Similarly, data channels require three sets × (5 sec for 1st data channel + 2 sec for 2nd and 3rd data channels) = 27 sec. Here, 5 sec and 2 sec come from our experimental results shown in Fig. 15. Additionally, it takes about 85 sec to complete PnP with our proposed fast search mechanism as shown in Fig. 14. The experimental link-up search time is, therefore, estimated to be 85 − (27 + 27) = 31 sec. In this experiment, one slot was set to 1 sec. The simulation results in Sect. 4.3 indicate that the proposed link-up search mechanism requires 35 slots, or 35 sec. These results show that the simulation results of our link-up search mechanism correspond to the experimental ones with a margin of error of only 11% or so. On the other hand, the required time for random search obtained from experiments (140 − (27 + 27) = 86 sec for 12-PnP search target interface OXC, 140 sec comes from Fig. 17) is shorter than the expected time for random search as indicated by the simulation (140 slots~140 sec, 140 slots, see Fig. 9). The reason is as follows. Figure 9 shows the expectation of required time to finish link-up search with a probability of 97.5%. As shown for Fig. 8, the matching probability against elapsed time has its peak on the left side (i.e., shorter elapsed time). Therefore, in the experiment, matching times are more likely to be short, which makes the experimental search time shorter than the simulated time, Fig. 9.

7. Conclusions

This paper established the optical PnP architecture to automatically construct GMPLS networks. The optical PnP architecture is mainly composed of optical PnP procedure, network and node (OXCs and routers) architecture, link-up search mechanism for OXCs, and LMP extension to exchange, between nodes, the information necessary for identifying adjacent nodes.

Optical PnP offers lower operation effort, misconfiguration avoidance, and fast network construction. Both simulation and experimental evaluations confirmed its performance and feasibility.

References

[1] E. Mannie, ed., "Generalized multi-protocol label switching (GMPLS) architecture," RFC 3945, Oct. 2004.

[2] D. Katz, K. Kompella, and D. Yeung, eds., "Traffic engineering (TE) extensions to OSPF version 2," RFC3630, Sept. 2003.

[3] K. Shimizu, R. Hayashi, I. Inoue, and K. Shiomoto, "Plug and play techniques for optical network configuration," ECOC2009, P5.10, Sept. 2009.

[4] K. Shimizu, R. Hayashi, I. Inoue, and K. Shiomoto, "Plug and play techniques for GMPLS network configuration," iPOP2009, 3-1, June 2009.

[5] R. Hayashi and K. Shiomoto, "Implementation of optical switches with plug-and-play mechanism in all-optical networks," OECC2010, 7A1-3, July 2010.

[6] R. Hayashi and K. Shiomoto, "Generalization and speed up of an optical plug play technique," Proc. IEICE Gen. Conf. 2010, BS-3-22, March 2010.

[7] J. Lang, ed., "Link management protocol (LMP)," RFC 4204, Oct. 2005.

[8] R. Hayashi and K. Shiomoto, "Draft-hayashi-ccamp-lmp-pnp-ext-00," internet-draft, July 2010.

[9] R. Hayashi and K. Shiomoto, "Optical plug and play implementation and its demonstration," iPOP2010, 1-1, June 2010.

[10] L. Berger, ed., "Generalized multi-protocol label switching (GMPLS) signaling resource reser vation protocol-traffic engineering (RSVP-TE) extensions," RFC3473, Jan. 2003.

[11] GMPLS-Engine: http://shop.ntt-at.co.jp/gmpls/6gmpls/

[12] JUNOS-SDK: http://www.juniper.net/us/en/products-services/nos/junos/junos-sdk/

[13] A. Doria, F. Hellstrand, K. Sundell, and T. Worster, eds., "General switch management protocol (GSMP) V3," RFC3292, June 2002.

Appendix

We assume that the random search cycle of both nodes is synchronized in the analytical model. We used the number of unmatched wavelength links between neighbor nodes as the system state of the neighbor discovery mechanism.

Let p_n denote the probability that at least one matching is found among n candidates.

$$p_n = 1 - 1/2! + 1/3! - 1/4! + \cdots + (-1)n^{+1}/n!. \quad \text{(A· 1)}$$

Let $q_{N,\eta}$ denote the probability that more than n matches are found among N candidates.

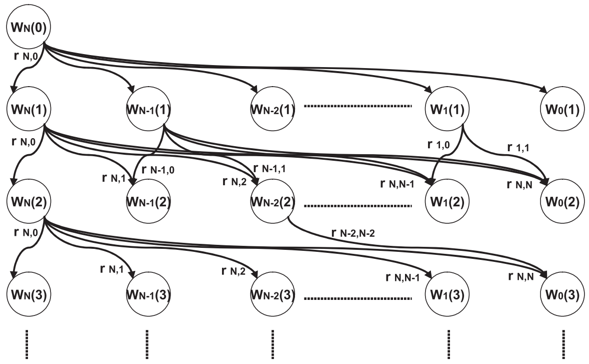

Fig. A· 1 Transition diagram (to 3rd cycle).

$$q_{N,\eta} = {}_N\mathrm{C}_n/N(N-1)\dots(N-(n-1)) = 1/n!. \quad \text{(A· 2)}$$

Let $r_{N,n}$ denote the probability that exactly n matches are found among N candidates.

$$\begin{aligned} r_{N,n} &= Prob(\text{no matches are found among } N\text{-}n \text{ candidates}) \\ &\quad * Prob(\text{more than } n \text{ matches are found among} \\ &\qquad N \text{ candidates}) \\ &= Prob(1 - Prob(\text{at least one match is found} \\ &\qquad \text{among } N\text{-}n \text{ candidates})) * q_{N,n} \\ &= (1 - p_{N-n}) * q_{N,n}. \end{aligned} \quad \text{(A· 3)}$$

Let $s_i(t)$ denote the probability that exactly t candidates remain unmatched after i cycles. Let $F_i(t)$ denote the probability that matching of all candidates finishes at the i-th cycle.

$$\begin{aligned} &F_i(t) = s_{i-1}(j1) * s_{i-2}(j2) * s_{i-3}(j3) * \dots * s_0(N) \quad \text{(A· 4)} \\ &0 < j_1 \leqq j2 \leqq j3 \leqq \cdots\dots \leqq N \end{aligned}$$

Here, "*" represents convolution. As shown in Fig. A· 1, where $w_N(i)$ denotes the probability that exactly N candidates remain unmatched after i cycles, each cycle has N states, and there are many combinations of states on each cycle, which are a route to the completely matched state.

Let $P(t)$ denote that the probability that it takes t slots to finish the matching at any cycle.

$$P(t) = \sum_{i=1}^{\infty} F_i(t) \quad \text{(A· 5)}$$

Based on the random search model above, our proposed search method is computed as follows. Here, N and M are the number of search target interfaces in a node, and the number of wavelengths multiplexed between neighbor nodes, respectively.

$$\begin{aligned} &\text{Search time with proposal (a)+(b)} \\ &\quad = \text{Search time required when the number of search} \\ &\qquad \text{target interface is } N/M + (M-1) \times (N/M) \end{aligned} \quad \text{(A· 6)}$$

N/M represents the number of WDM groups, or the number of control channels. This is due to the proposed method

(a) which identifies only control channel interfaces (instead of all of the interfaces) as the first search target. $(M-1)$ represents the number of data channels in a WDM group, and becomes the time required (slot) to search for them. A node has N/M WDM groups, so $(M-1) \times (N/M)$ becomes the time required to search for all data channel interfaces in a node.

Rie Hayashi received the B.S. and M.S. degrees in Electronic Engineering from the University of Tokyo, Tokyo, Japan, in 2001 and 2003, respectively. In 2003, she joined NTT (Nippon Telegraph and Telephone Corporation) Network Service Systems Laboratories, where she has been engaging in research of IP-optical traffic engineering techniques. She is now researching traffic engineering techniques for next-generation backbone network. She received the Young Engineer Award from the IEICE in 2009.

Kaori Shimizu was an Engineer, Broadband Network Systems Project, NTT Network Service Systems Laboratories, and currently she is working in NTT Communication. She received the B.S. degree from the University of Electro-Communications, Chofu, Tokyo in 1995. She joined NTT the same year.

Kohei Shiomoto is a Senior Research Engineer, Supervisor, Group Leader at NTT Network Service Systems Laboratories, Tokyo, Japan. He joined the Nippon Telegraph and Telephone Corporation (NTT), Tokyo, Japan in April 1989. He was engaged in research and development of ATM networks in NTT Laboratories. From August 1996 to September 1997 he was engaged in research in high-speed networking as a Visiting Scholar at Washington University in St. Louis, MO, USA. Since September 1997, he had been engaged in research and development in the areas of IP/GMPLS networking, IP and optical networking at NTT Network Innovation Laboratories and NTT Network Service Systems Laboratories. Since April 2006, he has been leading the IP Optical Networking Research Group in NTT Network Service Systems Laboratories. He received the B.E., M.E., and Ph.D. degrees in information and computer sciences from Osaka University, Osaka in 1987 1989, and 1998, respectively. He is a member of IEEE, and ACM. He received the Young Engineer Award from the IEICE in 1995. He received the Switching System Research Award from the IEICE in 1995 and 2001.