Title	A novel self-routing address scheme for all-optical packet-switched networks with arbitrary topologies
Author(s)	Yuan, XC; Li, VOK; Li, CY; Wai, PKA
Citation	Journal Of Lightwave Technology, 2003, v. 21 n. 2, p. 329-339
Issued Date	2003
URL	http://hdl.handle.net/10722/44861
Rights	Creative Commons: Attribution 3.0 Hong Kong License

A Novel Self-Routing Address Scheme for All-Optical Packet-Switched Networks With Arbitrary Topologies

X. C. Yuan, V. O. K. Li, *Fellow, IEEE*, C. Y. Li, *Member, IEEE*, and P. K. A. Wai, *Senior Member, IEEE, Member, OSA*

***Abstract*—Pure all-optical packet-switched networks in which both header processing and packet routing are carried out in the optical domain overcome the bandwidth bottlenecks of optoelectronic conversions and therefore are expected to meet the needs of next generation high speed networks. Due to the limited capabilities of available optical logic devices, realizations of pure all-optical packet-switched networks in the near future will likely employ routing schemes that minimize the complexity of routing control. In this paper, we propose a novel self-routing scheme that identifies the output ports of the nodes in a network instead of the nodes themselves. The proposed address scheme requires single bit processing only and is applicable to small to medium size pure all-optical packet-switched networks with arbitrary topologies. Unlike traditional self-routing schemes, multiple paths between two nodes can be defined. Hierarchical address structure can be used in the proposed routing scheme to shorten the address.**

***Index Terms*—Communication system routing, optical data processing, packet switching, photonic switching systems, wide-area networks.**

I. Introduction

A TYPICAL packet switch consists of a header processing unit (HPU) and a packet-routing unit (PRU). The HPU processes the header of an incoming packet, determines through which output port the input packet should be sent, and sets the PRU accordingly. In a pure all-optical packet switch, both the signal processing in the HPU and the packet routing in the PRU are carried out all-optically. Most of the current all-optical packet switches are in fact hybrid optical packet switches, i.e., while the packet remains in the optical domain, a copy of the packet header is converted into electrical signals for processing in the HPU [1], [2]. The decision of the HPU is then used to set the PRU to route the packet. Recent efforts on the hybrid approach include the realizations of the PRU using thermooptical [3], [4], electromechanical [5], and electrooptical means [6], [7]. Testbeds of hybrid all-optical packet-switched networks include the European Advanced Communications Technologies and Services (ACTS) Keys to Optical Packet Switching (KEOPS) Project, which uses fixed-size packets with headers transmitted at a lower bit rate and transparent payload [8]. More recently, the Information Society Technologies (IST) Data and Voice Integration Over DWDM (DAVID) Project investigates the use of multiprotocol label switching (MPLS)-based mechanisms to optimize the utilization of both electronic and optical resources [9], [10]. The hybrid approach is a compromise solution to the immediate demand for high-speed packet-switched networks. While it combines the flexibility of electronics and the huge transmission bandwidth of optics, the packet-switching rate is limited by the comparatively low bandwidth of electronic devices. The header processing overheads become significant as the transmission rate increases even if MPLS protocol is used [11]. Pure all-optical packet-switched networks eliminate such electronic bottleneck and are regarded as the ultimate networks of the future. Optically controlled PRUs have been demonstrated using the terahertz optical asymmetric demultiplexers (TOADs) [12]–[16] and the optical flip-flops [17]–[19]. However, much work have to be done before they become practical. Progress in all-optical implementation of the HPU is limited by the lack of practical optical buffers and the capabilities of available photonic devices [1], [2]. Current optical buffers are made from fiber delay lines which are bulky and the delays are fixed [20], [21]. For optical logic devices, only simple Boolean logic functions such as AND, OR, NOR, INVERT, and XOR have been demonstrated so far [22]–[25]. These optical-logic devices typically are bulky and integration is difficult. Complex optical-logic circuits are not feasible yet. Hence, pure all-optical packet-switched networks of the near future must adopt address schemes that simplify routing control and require single-bit processing only. All-optical implementation of traditional packet-switching strategies such as store-and-forward or table lookup would not be possible.

Self-routing has been used in packet and ATM switches to reduce the hardware cost and control complexity [26]. Upon entering a switch, a packet is attached with a self-routing address that contains all the routing information from all input ports to the desired output ports of the switch. Intermediate switching elements forward the incoming packet to the output ports using bit-by-bit processing of the self-routing address header. The additional header is removed once the packet reaches the desired output port. Much work has been carried out on switch structures and self-routing algorithms [11], [27]. The robustness and simplicity of self-routing also makes it a promising candidate

Manuscript received February 11, 2002; revised July 24, 2002. This work was supported in part by the Areas of Excellence Scheme, established under the University Grants Committee of the Hong Kong Special Administrative Region, China (Projects AoE/E-01/99), and a Grant from The Hong Kong Polytechnic University (Project Number A-PC68).

X. C. Yuan and V. O. K. Li are with the Department of Electrical and Electronic Engineering, The University of Hong Kong, Hong Kong (e-mail: xcyuan@eee.hku.hk; vli@eee.hku.hk).

C. Y. Li and P. K. A. Wai are with the Department of Electronic and Information Engineering, Photonics Research Center, The Hong Kong Polytechnic University, Hong Kong (e-mail: enli@polyu.edu.hk; enwai@polyu.edu.hk).

Digital Object Identifier 10.1109/JLT.2003.808755

for pure all-optical packet-switched networks [15], [28]. Traditional self-routing schemes, however, can only be applied to networks with regular topologies such as hypercube networks and the shufflenets [29]. For networks with arbitrary topologies, it is necessary to map the physical topologies of the networks to logical networks with regular topologies [30], [31]. Determination of the optimal mapping is an N-P complete problem [32]. Heuristics, such as genetic algorithm and simulated annealing, are required [32]. For a given network topology, it is hard in general to determine the most suitable regular topology for the mapping. Besides, the paths between nodes are fixed in self-routing schemes. It is therefore difficult to implement congestion control and traffic engineering. Rerouting of the paths for system reconfiguration is also a problem. In this paper, we propose a novel self-routing address scheme that is applicable to networks with arbitrary topologies. Only single-bit processing is required. The proposed scheme allows multiple addresses for the same node. Each address encodes a different set of paths from other nodes to this node, i.e., multiple paths between two nodes are possible. Multiple addresses of the nodes can be used to increase the reliability and flexibility of the system. The proposed self-routing address scheme can be readily adapted to a hierarchical structure for use in hierarchical networks. We focus on the address protocols in this paper. Investigation on packet-contention problems has been reported in [33].

The paper is organized as follows. In Section II, we describe two address structures of the proposed self-routing scheme and discuss the limitations set by the address lengths on the network sizes. Section III generalizes the proposed address scheme to hierarchical networks. In Section IV, we discuss all-optical implementations and the processing delay of the proposed self-routing addresses. In Section V, we consider reliability and scalability issues. Different methods to reduce the length of the address are discussed in Section VI. Section VII concludes the paper.

II. The Basic Self-Routing Scheme

In the proposed self-routing address scheme, each output port of all the nodes in a network is associated with a bit in the address header. Different output ports may be associated with the same address bit, but an output port will not be associated with more than one address bit. The HPU of a node processes the address of an input packet bit-by-bit and sets the PRU to route the packet to the output port with its corresponding address bit set to **1**. Depending on whether there is a one-to-one correspondence between the output ports and the address bits, we defined two self-routing address structures in Sections II-B and II-C.

For both of the proposed self-routing address structures, the address of a node encodes a unique path from any other node to the node itself. Since an address contains the instructions for which output port to use at every node irrespective of where the packets originate, two paths that intersect at an intermediate node must use the same output port at that node. This requirement is summarized in the following condition.

Condition 1: If the paths from two different nodes to the same destination node meet at an intermediate node, the subsequent links and nodes used by the two paths must be the same.

Condition 1 ensures that the routing instructions from different nodes to the same destination node can be encoded in one single address. If there are multiple paths between two nodes, the destination node will have multiple addresses, with each address encoding a different set of paths to the node. Some of the paths encoded in the different addresses of the same node can be the same. The paths encoded in each address must satisfy the previous condition. The routing information encoded in each address, for the same node or different nodes, are independent of one another. If a set of paths to a node does not satisfy Condition 1, multiple addresses can be used to encode all the paths using fictitious paths to complete the addresses. These fictitious paths are for address construction purpose and will not be actually used. The following example illustrates how to construct the addresses and how to use multiple addresses to encode a set of paths that do not comply with Condition 1.

In the following discussion, we consider a network made up of N nodes and K links. For simplicity, all links are assumed to be bidirectional. Thus, the number of input ports equals to that of the output ports. The proposed scheme can be applied to unidirectional links as well. Each node is arbitrarily labeled from 1 to N. The output ports of each node are also arbitrarily labeled from 1 to $d(i)$, where $d(i)$ is the number of output ports of the ith node. We have $\sum_{i=1}^{N} d(i) = 2K = ND$, where $D = 2\,\mathrm{K/N}$ is the average output degree of the nodes in the network.

A. Address Structure I

In this address structure, each output port of the nodes in a network has a one to one correspondence with a bit in the address. The address of a node therefore contains $L = 2K$ bits. We group the bits corresponding to the output ports of the same node to form N address subfields. Each subfield corresponds to one node in the network. The ith subfield of an address consists of $d(i)$ bits. All bits in the ith address subfield of node i are set to zero. For the jth address subfield of node i, $j \neq i$, the kth bit is set to **1**, where the path to node i exits node j through the kth output port. The other bits at the jth address subfield are set to zero. There is a total of $(N-1)$ **1** bits out of the L bits in each address.

When a node receives a packet, it only processes the address subfield corresponding to the node itself. A node recognizes that a packet has arrived at the destination if the corresponding address field is all zeros. Otherwise, it forwards the packet to the local output port as specified.

As an illustration of the self-routing address scheme and the use of multiple addresses, we consider the five-node six-link network shown in Fig. 1. The nodes are labeled from 1 to 5. We assume that there is a path between any two nodes. All paths are randomly selected. Altogether, 20 paths are defined. A path is represented by the sequence of nodes it uses. We represent the path from node i to node j as $P(i,j)$. The 20 paths are given in Table I.

We label the output ports of each node with numbers in parentheses as shown in Fig. 1. There are five address subfields in the node address corresponding to the five nodes in the network. The number of bits in each address subfield is given by

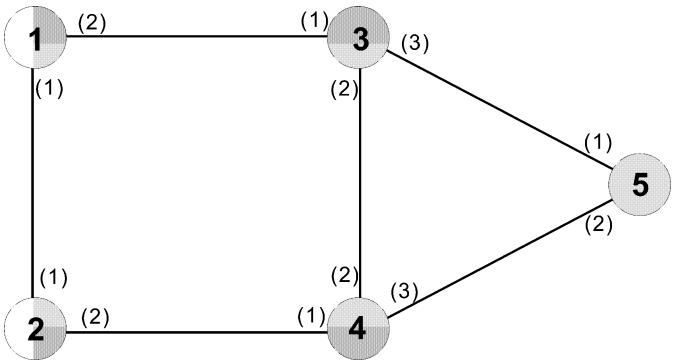


Fig. 1. Five-node six-link network. All links are bidirectional. A total of 20 paths are defined in Table I for connections between nodes.

TABLE I
TWENTY PATHS BETWEEN THE NODES FOR THE NETWORK SHOWN IN FIG. 1. THE PATH SETS $\{P(2,1), P(3,1), P(4,1), P(5,1)\}$ AND $\{P(1,5), P(2,5), P(3,5), P(4,5)\}$ DO NOT COMPLY WITH CONDITION 1

P(2,1) = 21	P(3,1) = 31	P(4,1) = 421	P(5,1) = 5431
P(1,2) = 12	P(3,2) = 312	P(4,2) = 42	P(5,2) = 542
P(1,3) = 13	P(2,3) = 213	P(4,3) = 43	P(5,3) = 53
P(1,4) = 124	P(2,4) = 24	P(3,4) = 34	P(5,4) = 54
P(1,5) = 135	P(2,5) = 2435	P(3,5) = 35	P(4,5) = 45

$d(1) = 2$, $d(2) = 2$, $d(3) = 3$, $d(4) = 3$, and $d(5) = 2$. The total number of bits $L = 12$.

The address of node 1 is constructed as follows. The bits in the first field are set to zero, i.e., **00.** For the second field, we look at the path $P(2,1) = 21$. Since a packet sent from node 2 to node 1 is transmitted through the output port labeled (1), the first bit of the second address field of node 1 is set to **1**, and the second bit of the second address field is set to **0**. The second address field in the address of node 1 is therefore **10.** Similarly, the third address field is **100.** The address of node 1 so far is **00_10_100_???_??.**

For the fourth address field, a packet sent from node 4 to node 1 is first routed from node 4 to node 2 through output port (1), and then from node 2 to node 1 through output port (1). From the first part of the routing instruction, the fourth address subfield is given by **100.** The packet is now at node 2. So we look at the second address subfield of the address. From the second part of the routing instruction, the second subaddress field in the address of node 1 should be **10**, which agrees with what has been put down earlier from the consideration of the path $P(2,1)$. This consistency is guaranteed since paths $P(2,1)$, $P(3,1)$ and $P(4,1)$ comply with Condition 1. However, path $P(5,1) = 5431$ for the fifth address subfield is in conflict with the contents of the first 4 address subfields because the four paths violate Condition 1. In path $P(5,1)$, after a packet is routed to node 4, it is then sent to node 3. However the packet should be sent to node 2 according to $P(4,1)$. To accommodate $P(5,1)$, we can either redefine $P(4,1)$ to 431, or we can modify $P(5,1)$ as 5421. The address of node 1 will be **00_10_100_010_01** (1a) for the former case and **00_10_100_100_01** (1b) for the latter case. If we are not able to modify the original paths, we may add the paths $P^*(4,1) = 431$ and $P^*(5,1) = 5421$. Node 1 now has two valid addresses, 1a and 1b, each encoding a different path. Note that in both addresses, the paths from node 2 or node 3 to node 1 are identical. The additional paths are for address construction purpose only. They may not be actually used for packet transmission. For example, if node 4 only uses address 1b and node 5 only uses address 1a, then the added paths, $P^*(4,1)$ and $P^*(5,1)$, will not be used.

TABLE II
SELF-ROUTING ADDRESSES FOR THE NODES IN THE NETWORK IN FIG. 1 AND THE PATHS SHOWN IN TABLE I. MULTIPLE ADDRESSES $\{1a, 1b\}$ AND $\{5a, 5b\}$ HAVE BEEN USED FOR NODES 1 AND 5, RESPECTIVELY

Address \ Field:	1	2	3	4	5
1a:	00	10	100	010	01
1b:	00	10	100	100	01
2:	10	00	100	100	01
3:	01	10	000	010	10
4:	10	01	010	000	01
5a:	01	01	001	010	00
5b:	01	01	001	001	00

Table II gives the self-routing addresses of the five nodes of the network corresponding to the routing paths shown in Table I. Seven addresses are constructed. Together they contain all the 20 paths chosen. Besides node 1, we have to assign two addresses to node 5 because the paths $P(1,5)$, $P(2,5)$, $P(3,5)$, and $P(4,5)$ do not comply with Condition 1.

Normally, a packet will not be trapped in a loop if the address is constructed properly. A node can easily test the validity of the addresses in its address table by reconstructing the paths from the addresses. However, the headers can still be corrupted even though the bit error rate of optical fibers is better than 10^{-9}. Some of these corrupted packets cannot exit the network because either the erred bits cause the packets to travel in loops or the erred bits are in the subfield of the destination node. Optical implementation of error correction algorithms for the headers at each node will require complex processing of the signals. Even a parity check is difficult to implement all-optically [34]. It will be simpler just to remove these packets based on their sojourn time in the networks by implementing a time-to-live (TTL) field [35]. An all-optical TTL scheme that requires only single-bit processing is described in [33].

B. Address Structure II

The address described in Section II-B has a two-tier structure. The address subfields identify the nodes, and the bits in the address subfield identify the output ports of the node. We note that when a packet is in node j, the bits not in the jth address subfield are not used at all. Since each address bit corresponds to a unique output port, the length of the address increases linearly with the total number of output ports and will affect the throughput and processing time. The length of the address can be significantly shortened if a bit is associated with more than one output port.

A simple address structure of this type is that each address bit identifies a unique node of a network instead of an output port. The length of the new address structure is N bits, where for simplicity, we assume that each node has only one address. The address of node i has its ith bit set to **1** and all other bits to **0**. We note that at any node j, the remaining $(N-1)$ nodes can be partitioned according to the output ports that packets destined to these nodes exit node j. Therefore, at each node j, we associate the kth bit where $k = 1, 2, \ldots, N$ with the output port specified in the routing instruction to node k. In other words, the address bits now have dual roles. It identifies both the destination node and the local output port leading to the destination node at each node. The association between the address bits and the nodes is global, while that between the address bits and the local output ports is node dependent. Compared to address structure I, which stores the routing information in the address, address structure II stores the routing information in the association between the address bits and the output ports at each node. Consequently, when the routing paths to a node change, only the addresses are modified in address structure I. The association between the address bits and the output ports of the nodes is not changed. In the address structure II, however, the addresses are not modified but the association between the bits and the output ports at the affected nodes must be changed.

Routing of a packet using address structure II is straightforward. When a node receives a packet, it checks the whole address header bit by bit. If the bit corresponding to the node is set to 1, the node retrieves the packet. Otherwise, the node routes the packet to the output port as indicated by the position of the **1** bit.

Table III gives an example of the self-routing addresses for address structure II for the network shown in Fig. 1 and the paths defined in Table I. Table IV gives the corresponding mapping between the bits in the address and the output ports at each node. Both address structures I and II require single bit processing. The choice between the two address structures for a given network depends on the network topology, the relative ease in the implementation of each address structure, and the lengths of the addresses.

TABLE III
ALTERNATE SELF-ROUTING ADDRESSES FOR THE NODES IN THE NETWORK SHOWN IN FIG. 1 AND THE PATHS DEFINED IN TABLE I

Address \ Bit position:	1	2	3	4	5	6	7
1a:	1	0	0	0	0	0	0
1b:	0	1	0	0	0	0	0
2:	0	0	1	0	0	0	0
3:	0	0	0	1	0	0	0
4:	0	0	0	0	1	0	0
5a:	0	0	0	0	0	1	0
5a:	0	0	0	0	0	0	1

TABLE IV
MAPPING BETWEEN THE BITS IN THE ADDRESS STRUCTURE II WITH THE OUTPUT PORTS AT EACH NODE

Node \ Bit position:	1	2	3	4	5	6	7
1:	-	-	1	2	1	2	2
2:	1	1	-	1	2	2	2
3:	1	1	1	-	2	3	3
4:	2	1	1	2	-	2	3
5:	2	2	2	1	2	-	-

C. Lengths of the Addresses

In address structure I, since we associate a unique address bit to every output port of the nodes in a network, the address is $L = ND$ bits long. For comparison, traditional addresses that label the nodes instead of the output ports is only $\log_2 N$ bits long. In other words, elimination of the all-optical routing table lookup function and simplification of processing requirement are done at the expense of increase in the address length. Obviously, the address length cannot be too long otherwise the throughput will reduce and the header processing time will increase. The maximum address length in turn set a limit on the network size that it is beneficial to use address structure I. For data protocols, the overheads are typically less than 5% [36]. We note that the maximum payload size of Internet Protocol (IP) packets is 65 536 bytes. There is an option in $IPv6$ to extend the payload size beyond this limit. Therefore if the maximum IP packet size is used as slot length, a 1% address overhead means an address length limit of about 5 kb long.

The header processing time for the proposed addresses is expected to be on the order of the header length because we assume on-the-fly optical processing. For current hybrid optical packet-switched networks, the headers are converted into electrical signals either directly using ultrafast photodetectors or by optically demultiplexing first into individual bits before detection. The electrical signals are then processed to determine the required output port which in general requires doing a routing table lookup. The information is then used to set an electrooptical switch to complete the routing action. Typically, the response time of detectors is several picoseconds [37], the packet processing time is a few microseconds [38], and the switching time of a lithium niobate electrooptical switch is tens of picoseconds [39]. The address length, and hence the required header processing time, of address structure I, should therefore be much less than one microsecond for the scheme to be an attractive alternative to hybrid optical packet-switched networks. In other words, the address should be on the order of 1 kb or less for a line rate of 10 Gb/s, 4 kb or less for 40 Gb/s, and so on.

The limitation on the address length will be relaxed when optical processing technology matures beyond the single-bit logic gates. For example, if binary encoding of the address is allowed without significant increase in processing time, the address will be only $\log_2 N + \log_2 D$ bits long. Address structure I can then be applied to much larger networks. In the near future, however, applications of address structure I will be limited to small to medium size networks such as in backbone networks. The size of current backbone networks is in general not large. For example, there are 27 nodes in AT&T Internet Protocol service backbone network, and 31 nodes for European research back-

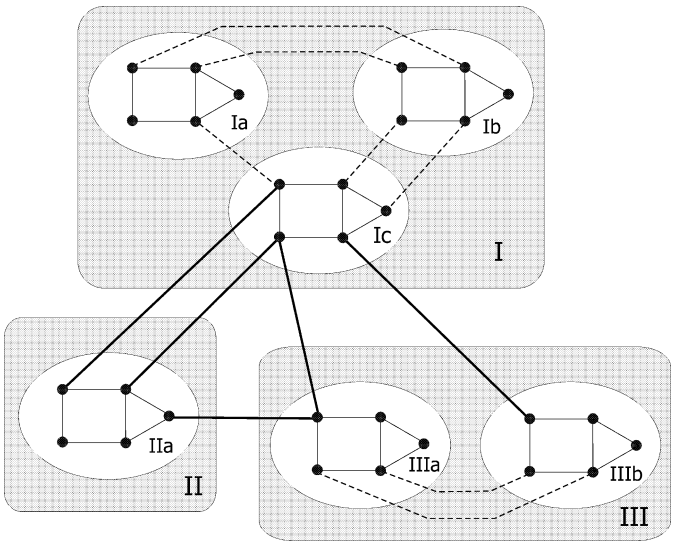


Fig. 2. Three-level hierarchical network.

bone network Multi-Gigabit pan-European Research Network (GEANT) [40], [41]. The average node output degree in the these networks is about three. The address length using address structure I will be about a hundred bits only.

The length of address structure II equals the number of nodes in the networks, and is in general much shorter than that of address structure I. Thus, address structure II can be applied to much larger networks. The main drawback of structure II, however, is that the updating of routing paths is more difficult than that of structure I. To modify the routing path in address structure I, we only need to change the position of the 1 bit in the address field. No modification of the mapping between the bit position and the output ports in the intermediate nodes is required. In structure II, the same address is used even if the routing paths are changed. We have to change the bit position to output port mapping in the intermediate nodes. An overlay network may be required to update the routing paths. Thus address structure II should be used in networks in which modifications of routing paths are rare.

The proposed address structures can in principle be applied to wavelength-division-multiplexed networks if optically controlled wavelength converters are available. Each wavelength can be considered to define a different network. The *W* networks, where *W* is the number of wavelengths, are interconnected through nodes that have partial or full wavelength conversion capabilities. The number of nodes and links in effect increase *W* folds. The addresses are therefore *NDW* bits and *NW* bits long for address structure I and II, respectively. This further constrains the network size for address structure I. Thus address structure I is not suitable for wavelength division multiplexed networks unless the number of wavelengths is small or optical processing technology improves.

III. Hierarchical Addresses

Hierarchical addressing can be used in the proposed self-routing scheme in order to shorten the length of address structure I. Since not all the output ports can be uniquely represented by a bit in a hierarchical address, some bits in the address must be associated with multiple output ports as in the case of address structure II. In the following, we will describe a hierarchical self-routing address scheme for networks with arbitrary topologies.

Before constructing a hierarchical address, we need to organize a network into multilevels. First, the nodes are grouped to form different subnetworks. Each node belongs to one subnetwork only. The subnetworks are denoted as level-2 nodes, while the original nodes are classified as level-1 nodes. The level-2 nodes interconnect with one another forming a network with a topology different from the original one. An output port of a level-1 node that connects to another level-1 node of the same subnetwork is classified as a level-1 output port. An output port that connects to a level-1 node of another subnetwork, i.e., a level-2 node, is classified as a level-2 output port. The level-2 nodes can be grouped together to form level-3 nodes. The process is repeated until the required levels of hierarchy are reached. In the following, a level-n node may sometimes be referred to as a Level-$(n-1)$ subnetwork, depending on the context of the discussion.

Fig. 2 shows a network of thirty nodes arbitrarily grouped into a three-level hierarchical structure. The nodes at levels 1, 2, and 3 are represented by dots, circles, and rectangles, respectively. The connections at levels 1, 2, and 3 are represented by thin solid lines, dashed lines, and thick solid lines, respectively. Fig. 2 shows six level-1 subnetworks, three level-2 subnetworks, and one level-3 subnetwork. The topology of every level-1 subnetwork is the same as that in Fig. 1. The topologies of the level-2 and level-3 subnetworks are shown in Fig. 3. A subnetwork may consist of only one node and there may be multiple connections between two nodes. The numbers in parenthesis are the local labels of the output ports for address construction purpose. The nodes at levels 2 and 3 are labeled as shown in Fig. 2. The labels used for a level-n node and its level-$(n-1)$ subnetwork content is chosen to be the same for convenience.

Packet routing in a hierarchical network is designed to be carried out hierarchically starting from the highest level subnetwork. The hierarchical address of a level-1 node in an M-level hierarchical network contains M subaddresses. The level-n subaddress contains the routing instruction in the level-n subnetwork, or equivalently the level-$(n+1)$ node, to which the destination level-1 node belongs. Using address structure I, we associate each level-n output port in the level-n subnetwork to which the destination level-1 node belongs to a unique bit in the level-n subaddress.

Since a level-n node is made up of level-1 nodes, the input port and the output port that a packet enters and leaves a level-n node in general do not belong to the same level-1 node. It is therefore necessary to route the packet from the level-1 node that the packet enters the level-n node to the level-1 node that the packet leaves the level-n node. To do so, we use address structure II to associate a set of paths from every level-1 node that made up the level-n node to each of the level-n output port with the bits chosen earlier to represent these level-n output ports in the level-n subaddress. Each set of paths satisfies Condition 1. In other words, the address bits in the address subfield of a level-n node now also serve as the "internal" address of the level-n output ports. When these bits are set to **1**, a packet placed

at any level-1 nodes inside the level-n node will be forwarded to the corresponding level-n output ports. A level-n output port may have multiple addresses similar to the multiple addresses of a node as discussed in Section II. The address field corresponding to the destination level-n node is set to zero. Only one bit in each of the rest of the address fields is set to **1**, indicating the level-n output port that will be used for that node.

The routing of a packet is carried out as follows. When a level-1 node receives a packet, it first checks the level-M subaddress in the address. Specifically, the node checks the address field of the level-M node to which the node belongs. If the field is nonzero, the packet does not belong to the level-M node that contains the level-1 node. The node then forwards the packet to one of its output ports according to the position of the **1** bit in this field. If the address field is zero, the packet is intended for the level-M node to which the level-1 node belongs. The node then checks the level-$(M-1)$ subaddress and routes the packet accordingly. A node retrieves a packet if the corresponding address fields at all the M subaddresses are zero.

As an illustration, we use the three level network shown in Fig. 2 as an example. The hierarchical address of the nodes in the network consists of three parts arranged as **[level-3 subaddress][level-2 subaddress][level-1 subaddress]**. Each subaddress contains the routing instructions at each level of the network. The subaddress at each level is constructed in the same way as described in Section II using the topologies shown in Figs. 1 and 3. For example, we consider the level-3 subaddress. From Fig. 3, it consists of three address fields corresponding to the three level-3 nodes as **[(node I)_(node II)_(node III)]**.

In Fig. 3, we assume that there is only one set of paths for each level-3 output port, the address fields for level-3 nodes I, II, and III contain 4, 3, and 3 bits, respectively. An example of a level-3 subaddress is **[0000_100_010]**, which indicates that the destination level-1 node belongs to node I. From node III, a packet will be sent to node I through output port (2). From node II, the packet will be sent to node III through output port (1), and then to node I via output port (2) of node III.

The subaddresses at level-2 and level-1 are constructed similarly. The length of the subaddresses at the same level varies depending on the number of nodes in that subnetwork. From Fig. 3, the level-2 subaddresses of subnetworks I, II, and III have 10, 0, and 4 bits, respectively.

Finally, an example of a valid nodal address in the network is **[0000_100_010][010_00_010][01_01_001_010_00]**, which is an address of node 5 at subnetwork Ib. Address 5a in Table II is used as the level-1 subaddress. The level-2 subaddress contains the routing instruction at subnetwork I because the destination node belongs to node I as indicated in the level-3 subaddress. Another example of a valid address is **[1000_000_001][_][01_10_000_010_10]**. It is the address of node 3 at subnetwork IIa. The level-2 subaddress is empty because subnetwork IIa contains only one node. We note that without using hierarchical structure, the addresses of the network are 96 bits and 30 bits long using address structures I and II, respectively. The length of the two hierarchical addresses above are 30 and 22 bits.The amount of reduction in address length using hierarchical addresses depends on the the number

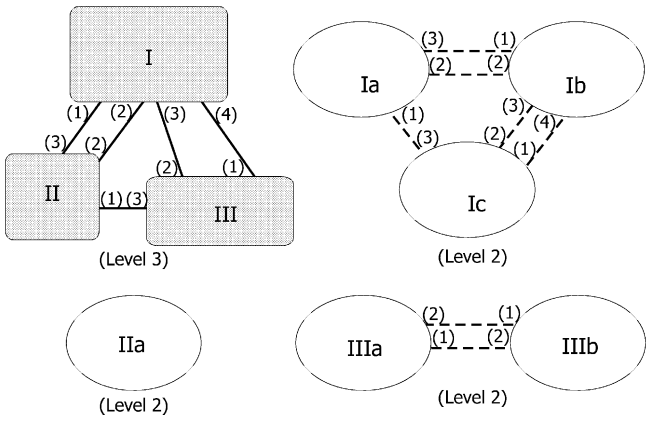


Fig. 3. Topologies of levels 2 and 3 subnetworks in Fig. 2.

of hierarchical levels M chosen and the partitions of the nodes into subnetworks at each level. To estimate the optimum reduction factor that can be achieved for a network of N nodes, we assume that every level-n node contains the same number X of level-$(n-1)$ nodes, where $n = 2, \ldots, M$. We have

$$\begin{aligned} N &= X^M \\ L_H &\leq XMD_{\max} \end{aligned} \tag{1}$$

where L_H is the length of the hierarchical address and $D_{\max}$ is the maximum node output degree of all the M levels of subnetworks. The bound for L_H is minimized if the number of hierarchical levels $M = \ln N$. In this case, $L_H \leq eD_{\max} \ln N$, where e is the natural number. Therefore, hierarchical addresses can be used to extend the limit on the length of the address structure I by a factor of at least $ND/eD_{\max} \ln N$.

While the use of hierarchical addresses can shorten the address length of address structure I, network management becomes more complicated. To modify the routing paths, the address as well as some of the mappings between the address bits and the nodes will have to be changed. Since every level-n node is composed of level-1 nodes, addition or removal of a link or a node may affect all the M levels of subaddresses.

IV. Implementation

In recent years, there have been much research on ultrafast, nonlinear, all-optical packet switching devices. The devices that are studied the most are based on ultrafast nonlinear interferometers (UNI) such as the Mach–Zehnder interferometer and the Sagnac interferometer. Early work concentrated on the exploitation of the optical Kerr effect as the switching nonlinearity [42]–[46]. However, fiber-based switches are bulky and not easily integrated. More recently, compact and integrated devices operating on the same principle using the nonlinear gain saturation of semiconductor optical amplifiers (SOAs) have been demonstrated [47], [48]. Bitwise logic such as OR, AND, INVERT, and NOR have been shown. The XOR operation is particularly important for network functions including address and header recognition, data encoding, and encryption. All optical XOR has been demonstrated at 100 Gb/s using nonlinear fiber loop mirrors [49], and at 20 Gb/s using SOA-based UNIs [24].

All-optical packet routing is difficult to realize. One all-optical device capable of performing all-optical packet routing as well as header processing is the TOAD, which is an SOA-based UNI [12]. A TOAD composes of a nonlinear optical fiber loop mirror which uses an SOA as the nonlinear element. The location of the SOA is offseted by Δx from the midpoint of the loop. A TOAD functions as a bitwise header processor if $\Delta x = \tau c/2$, where c is the speed of light in the fiber and τ is the bit interval. The TOAD can also function as an all-optically controlled packet router if $\Delta x = Tc/2$ instead, where T is the packet length. A 1×2 all-optical packet-routing node using single bit processing and an address header of the type proposed in this paper has been demonstrated in [14]. The 1×2 node is constructed from two TOADs. The address consists of a single bit only. One TOAD acts as an all-optical routing controller the output of which sets the state of the second TOAD which serves as an all-optical packet router. Depending on the value of the address bit, the packet is routed to one of the two output ports of the switch. The bit-rate is 250 Gb/s and the bit-error-rate performance of the switch is better than 10^{-9}. The packet length is limited by the recovery time of the SOA which is typically 0.4 ns. Beyond the recovery time of the SOA, the switching window of the TOAD decays exponentially [50]. Reference [16] discussed different ways to achieve a flat and uniform switching window.

Fig. 4 shows the schematic of a $1 \times k$ node implementing address structure I using the 1×2 switches ($S_i s$) demonstrated in [14]. An incoming packet is splitted into all the k output ports. The ith output port of the $1 \times k$ node is controlled by S_i. In each S_i, one output port is blocked while the other allows a packet to pass. The first TOAD in each S_i is synchronized with the ith bit of the address subfield corresponding to the node using a tunable delay line [51]. The recovered clock pulse in the first TOAD of each S_i is used as the control pulse to the first TOAD. Each S_i is set such that a packet is coupled to the port that allows a packet to pass if the associated address bit is 1. Otherwise the packet is sent to the port that blocks. Each S_i can be set to process a different address bit by adjusting its tunable delay line.

Implementation of address structure II is more complex as multiple bits in the address are associated with the same output port. One possibility is to first optically demultiplex a copy of the address header into individual bits. The bits associated with the same output port are synchronized and combined using tunable delay lines. The combined signal is then input as the address bit for the first TOAD in the corresponding S_i. Changing the association between the address bits and the output ports can be done by changing the delays. The demultiplexing of the address into individual bits however severely limits the address length, hence, the number of nodes in the network.

In address structure I, the processing delay in a node include the time the HPU takes to reach the node's address subfield and the processing time of the address. Depending on the positions of the node's address field in the header, the HPU may have to wait as long as the header duration before it reaches the target header bits. In address structure II, the maximum delay occurs if a node checks the whole address bit-by-bit sequentially until it reaches a none-zero bit. The processing time of the TOAD depends on the length of the loop mirror. Compact TOADs constructed from discrete components can have loop lengths of less than 1 m giving a delay of less than 5 ns. This delay is much longer than the recovery time of the SOA. Thus the switching window, hence the packet length, must be lengthened substantially in order for this technology to be practical. The total processing time of future all-optical packet-switching nodes should be only a small fraction of the packet length.

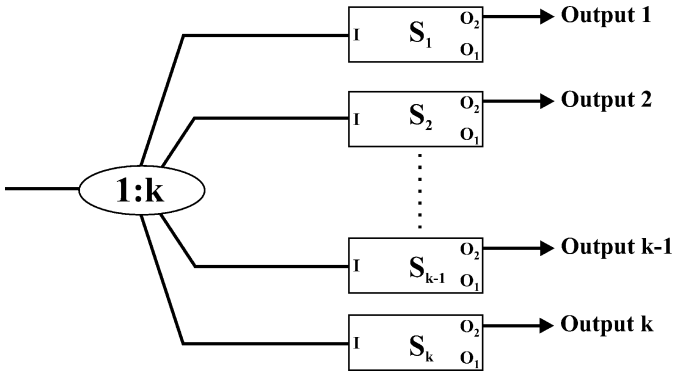


Fig. 4. A $1 \times k$ all-optical node. The $S_i s$ are 1×2 all-optical packeting routing nodes demonstration in [14]. One of the two output ports of each S_i, O_1, in this case is blocked. The ellipse is a 1 by k coupler.

V. Reliability and Scalability

Reliability deals with the robustness of the routing scheme in the event of link and/or node failures, while scalability is concerned with the increase in the complexity of the scheme when the network size increases. The proposed scheme uses fixed routing and inherits the disadvantages of fixed routing algorithms. However, unlike traditional self-routing schemes, the address of a node need not be unique. Multiple addresses of a node can be defined to encode multiple paths between nodes. For example, one can define two addresses of a node such that the two paths encoded are disjoint. Then if one of the paths fails, the source node can switch to a different path by using the other address. Address selection can also be based on congestion information, link utilization, and the required quality of service. This simplifies traffic engineering and reduces service interruption.

If multiple nodes and links fail, such that none of the paths encoded in all the addresses of a destination node is available to a source node, the addresses of the destination node can be recomputed to contain new routing information. Recomputation of the addresses of a node requires global knowledge of the network structure. This can be done either centrally or at each node. The new addresses will then be broadcasted to all nodes to update their address tables. Note that only the addresses of the nodes which use the failed nodes and links need to be modified because the routing information encoded in each address is independent. The network recovery time depends on the time to compute a new address, the propagation delay for the address broadcast, and the update of the address tables. Since an address of a node can be constructed even if only one path to the node exists, the recovery time depends mainly on the propagation delay.

In general, adding a node or a link will require address update and system reconfiguration which will disrupt services. To

minimize the disruptions, additional bits can be reserved in the header for assignment to new nodes or links. The extra unassigned bits will not affect the routing scheme. An advantage of this approach is that the existing addresses are still valid for routing packets when the new addresses with these extra bits assigned are sent to each node. Hence, there is no disruption in service when nodes or links are added. When nodes or links are removed from the network, new addresses will be sent to all the nodes. After the traffic stops using the nodes and links in question, these nodes and links can be safely taken down without affecting service.

The use of reserved bits, however, lengthens the addresses and hence the header processing time. The tradeoff depends on the difficulties in system reconfiguration, i.e., the re-assignment of address bits to the output ports.

VI. Address Compression

For the address structure I, the bits in different positions of the header often have identical values in all the addresses defined. In these situations, we can reduce the number of header bits with a slight modification of the self-routing scheme and no increase in hardware complexity. If we allow one of these identical bits to be shared by the other nodes, we can then eliminate the redundant bits and shorten the length of the address. In order for the self-routing to work correctly, we have to change the assignment of address bits to the output ports for the affected nodes.

As an illustration, we again consider the network in Fig. 1. For simplicity, we assume that each node has a unique address as shown in Table V. We note that column 5 and column 8, columns 7 and 10, and columns 9 and 11 are identical. We can then shorten the address field from 12 bits to 9 bits by eliminating the eighth, tenth, and eleventh bits.

Besides identical bits, we can also eliminate one of the two bits if the two bits complement each other in all addresses provided that an optical inverter is implemented. From Table V, columns 2 and 12 are complements of each other, so the length of the address can be reduced from 9 to 8 bits by eliminating the twelfth bit.

The resulting addresses are given in Table VI. In the compressed address format, the bit positions corresponding to the output ports of nodes 1 to 5 in ascending order of the output ports in each node are (1,2), (3,4), (5,6,7), (5,8,7), and (8, $\overline{2}$) respectively, where $\overline{2}$ is the complement of the bit at the second bit.

Further compression of the address may be possible. We observe that in the proposed routing scheme, a node routes an incoming packet to the output port specified in the address when it detects a **1** at the appropriate bit position. If the node processes its header bits sequentially, the values of the bits after the **1** in an address field is irrelevant. One may therefore increase the number of identical and complement address bits by changing some of these bits from zeroes to ones. However, one must ensure that in the compressed address, these bits are still processed after the bits with **1** in the address field. Otherwise, the routing instructions encoded in the addresses may be altered. For example, by changing the first two bits in column 6 of Table V into **1**, columns 2 and 6 are complementary to each other. It may appear that the sixth bit can be replaced by the second bit. But since the second bit is processed before the fifth bit in the address, the routing instructions encoded in compressed addresses of nodes 1 and 2 will be altered from that in the uncompressed ones.

TABLE V
The Self-Routing Node Addresses for the Network in Fig. 1. Each Node has a Unique Address

Field:	1	2	3	4	5
Address					
1b:	00	10	100	100	01
2:	10	00	100	100	01
3:	01	10	000	010	10
4:	10	01	010	000	01
5b:	01	01	001	001	00

TABLE VI
The Compressed Self-Routing Node Addresses for the Network in Fig. 1

Address								
1b:	0	0	1	0	1	0	0	0
2:	1	0	0	0	1	0	0	0
3:	0	1	1	0	0	0	0	1
4:	1	0	0	1	0	1	0	0
5b:	0	1	0	1	0	0	1	0

In the above example, we achieve 33% reduction of the address length by sharing bits among different nodes. The percentage of reduction is expected to decrease if multiple addresses are used. For example, there is no identical column for the seven addresses shown in Table II. Columns 2 and 12 are still complementary to each other. The address can be reduced from 12 to 11 bits, an 8% reduction in length only. Fig. 5 shows the average address lengths L using address structure I as a function of the number of nodes and connection probabilities. We use a pure random model to generate the different network topologies [52]. The connection probability between any two nodes is a constant p. The labels on the curves in Fig. 5 denote different connection probabilities. The nodes are connected using the shortest paths. Each node has a single address. One thousand random network topologies are generated for each choice of N and p. We observe that the average address length L grows with N^2 and p. Fig. 6 shows the average address length reduction factor R as a function of the number of nodes N. Average reduction factor R is defined as the average length of original addresses divided by the average length of compressed addresses. For address compression, we only remove a bit when either two bits are identical or complementary to each other in all addresses. No bit value is changed from **0** to **1**.

For a given connection probability and number of nodes, the expected node-output degree $\bar{d} = (N-1)p$ is the average number of output links connected to a node. In Fig. 6, we only

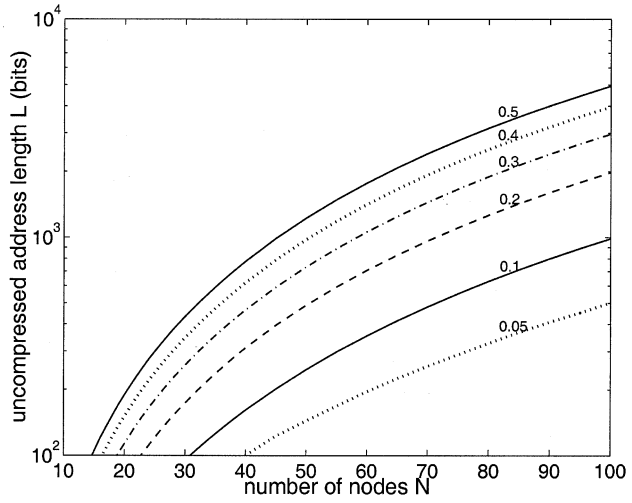


Fig. 5. Average uncompressed address length versus the number of nodes N. One thousand randomly generated network topologies are used to obtain each data point. The connection probabilities p between any two nodes are constant; $p = 0.05$, 0.1, 0.2, 0.3, 0.4, and 0.5.

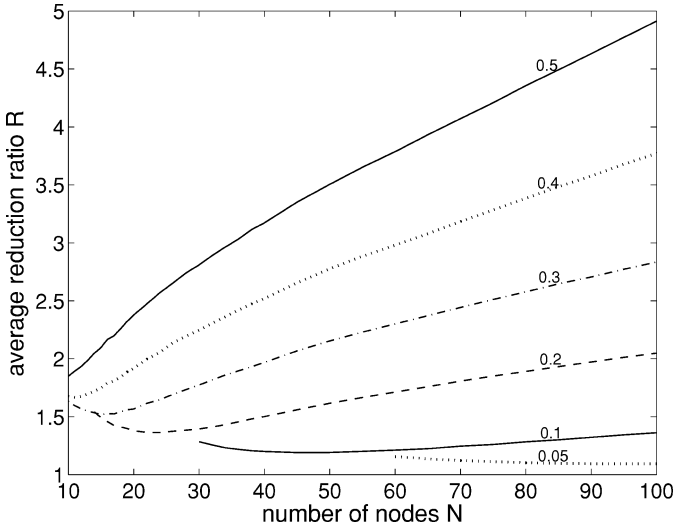


Fig. 6. Average address length-reduction factor R versus the number of nodes N. The conditions and notations of the curves are the same as that of Fig. 5.

include the results for $\bar{d} \geq 3$. When $\bar{d} \leq 3$, the network consists mainly of regular topologies such as buses, rings, and stars. Recall that the average node output degree of current backbone networks is about 3, giving a connection probability of about 0.1 [40], [41]. We observe that the average reduction factor R for $p = 0.05,\ 0.1, \ldots,\ 0.4$ initially decreases when N increases and reaches a minimum at $\bar{d} \approx 4.5$. R then increases as the number of nodes increases, or equivalently the average node output degree increases. When $\bar{d}$ is small, the paths coded in different addresses share many common links. The chance of having identical or complement bits in the addresses is high. The average reduction factor R is therefore larger than 1. When the number of nodes increases, the number of available paths between two nodes increases. The chance of identical or complementary bits decreases and hence R decreases.

The average uncompressed address length is given by $N\bar{d}$. When the number of nodes increases by 1, the average address length increases by $2(\bar{d}+p)$. Only one of these new bits will be filled with a **1**, the others will be filled with **0**'s. The chance of having identical and complementary bits in the addresses therefore increases with N. Consequently, the average reduction factor R also increases with the number of nodes. Similarly, R increases with the connection probabilities for the same N. From Fig. 6, the average address length is reduced by almost 80% using the simple compression technique for $p = 0.5$ and $N = 100$.

We note that the address-compression techniques can be applied to each part of the hierarchical address scheme. Finally, while compression can substantially shorten the address length, modification of routing to a node may cause address reconstruction of a large number of nodes because now a bit may be associated with a number of nodes. Furthermore, packets are more susceptible to loss caused by bit errors in the headers after address compression.

VII. Conclusion

In this paper, we propose a self-routing address scheme for all optical packet-switched networks. Despite its simplicity, the proposed scheme has a number of advantages. It can be implemented in arbitrary network topologies. There is no restriction on the choices of routing protocols. The paths between any two nodes can be chosen arbitrarily. Multiple paths between nodes are permitted by assigning multiple addresses to the nodes. The multiple addresses of a node can be used in alternate routing schemes in case of network failure or network congestion. One can reduce the chance of networkwide address reconfiguration for node and link addition by the use of extra unassigned address bits. We have demonstrated that the proposed scheme can be adapted to a hierarchical address structure for use in hierarchical networks or to shorten the addresses. We also showed that the address length can be shortened without increase in complexity of the scheme by sharing bits among different nodes of the network. Finally, because of its simplicity, the proposed scheme requires only single-bit optical processing. The length of the address, however, will limit the proposed self-routing address scheme to small-sized and medium-sized networks in the near future.

References

[1] V. W. S. Chan, K. L. Hall, E. Modiano, and K. A. Rauschenbach, "Architectures and technologies for high-speed optical data networks," *J. Lightwave Technol.*, vol. 16, pp. 2146–2168, Dec. 1998.

[2] A. Rodriguez-Moral, P. Bonenfant, S. Baroni, and R. Wu, "Optical data networking: Protocols, technologies, and architectures for next generation optical transport networks and optical internetworks," *J. Lightwave Technol.*, vol. 18, pp. 1855–1870, Dec. 2000.

[3] M. S. Yang, Y. O. Noh, Y. H. Won, and W. Y. Hwang, "Very low crosstalk 1×2 digital optical switch integrated with variable optical attenuators," *Electron. Lett.*, vol. 37, no. 9, pp. 587–588, 2001.

[4] T. Watanabe, T. Goh, M. Okuno, S. Sohma, T. Shibata, M. Itoh, M. Kobayashi, M. Ishii, A. Sugita, and Y. Hibino, "Silica-based PLC 1×128 thermo-optic switch," in *Proc. Eur. Conf. Optical Communications*, vol. 2, Amsterdam, The Netherlands, 2001, pp. 134–135.

[5] T. W. Yeow, K. L. E. Law, and A. Goldenberg, "MEMS optical switches," *IEEE Commun. Mag.*, vol. 39, pp. 158–163, Nov. 2001.

[6] B. C. Qiu, Y. H. Qian, O. P. Kowalski, A. C. Bryce, J. S. Aitchison, R. M. De La Rue, J. H. Marsh, M. Owen, I. H. White, R. V. Penty, A. Franzen, D. K. Hunter, and I. Andonovic, "Fabrication of 2×2 cross-point switches using a sputtered SiO_2 intermixing technique," *IEEE Photon. Technol. Lett.*, vol. 12, pp. 287–289, Mar. 2000.

[7] S. S. Saini, F. G. Johnson, D. R. Stone, W. Zhou, H. Shen, and M. Dagenais, "A 2 × 2 crosspoint switch fabricated on the passive active resonant coupler (PARC) platform," *IEEE Photon. Technol. Lett.*, vol. 13, pp. 203–205, Mar. 2001.

[8] P. Gambini *et al.*, "Transparent optical packet switching: Network architecture and demonstrators in the KEOPS project," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 1245–1259, July 1998.

[9] A. Banerjee *et al.*, "Generalized multiprotocol label switching: An overview of signaling enhancements and recovery techniques," *IEEE Commun. Mag.*, vol. 39, pp. 144–151, July 2001.

[10] [Online]. Available: http://david.com.dtu.dk/

[11] D. J. Blumenthal, P. R. Prucnal, and J. R. Sauer, "Photonic packet switches: Architectures and experimental implementations," *Proc. IEEE*, vol. 82, pp. 1665–1667, Nov. 1994.

[12] J. P. Sokoloff, P. R. Pruncal, I. Glesk, and M. Kane, "A terahertz optical asymmetric demultiplexer," *IEEE Photon. Technol. Lett.*, vol. 5, pp. 787–790, June 1993.

[13] I. Glesk, J. P. Solokoff, and P. R. Prucnal, "All-optical address recognition and self-routing in a 250 Gbit/s packet switched network," *Electron. Lett.*, vol. 30, no. 16, pp. 1322–1323, 1994.

[14] I. Glesk, K. I. Kang, and P. R. Prucnal, "Demonstration of ultrafast all-optical packet routing," *Electron. Lett.*, vol. 33, no. 9, pp. 794–795, 1997.

[15] P. Toliver, I. Glesk, R. J. Runser, K.-L. Deng, B. Y. Yu, and P. R. Prucnal, "Routing of 100 Gb/s words in a packet-switched optical networking demonstration (POND) node," *J. Lightwave Technol.*, vol. 16, pp. 2169–2180, Dec. 1998.

[16] R. J. Runser, D. Zhou, C. Coldwell, B. C. Wang, P. Toliver, K. Deng, I. Glesk, and P. R. Prucnal, "Interferometric ultrafast SOA-based optical switches: From devices to applications," *Opt. Quantum Electron.*, vol. 33, pp. 841–874, 2001.

[17] M. T. Hill, H. de Waardt, G. D. Khoe, and H. J. S. Dorren, "All-optical flip-flop based on coupled laser diodes," *IEEE J. Quantum Electron.*, vol. 37, pp. 405–413, Mar. 2001.

[18] M. T. Hill, A. Srivatsa, N. Calabretta, Y. Liu, H. de Waardt, G. D. Khoe, and H. J. S. Dorren, "1 × 2 all optical packet switch," in *Proc. Eur. Conf. Optical Communications*, vol. 4, Amsterdam, The Netherlands, 2001, pp. 546–547.

[19] Y. Liu, M. T. Hill, H. de Waardt, and H. J. S. Dorren, "All-optical switching of packets for all-optical buffering purposes," in *Proc. Eur. Conf. Optical Communications*, vol. 3, Amsterdam, The Netherlands, 2001, pp. 310–311.

[20] R. Langenhorst, M. Eiselt, W. Pieper, G. GroBkopf, R. Ludwig, L. Kuller, E. Dietrich, and H. G. Weber, "Fiber loop optical buffer," *IEEE J. Lightwave Technol.*, vol. 14, pp. 324–335, Mar. 1996.

[21] D. K. Hunter, M. C. Chia, and I. Andonovic, "Buffering in optical packet switches," *J. Lightwave Technol.*, vol. 16, pp. 2081–2094, Dec. 1998.

[22] K. L. Hall and K. A. Rauschenbach, "100-Gbit/s bitwise logic," *Opt. Lett.*, vol. 23, pp. 1271–1273, 1998.

[23] Y. Wang and J. Liu, "All-fiber logical devices based on the nonlinear directional coupler," *IEEE Photon. Technol. Lett.*, vol. 11, pp. 72–74, Jan. 1999.

[24] C. Bintjas, M. Kalyvas, G. Theophilopoulos, T. Stathopoulos, H. Avramopoulos, L. Occhi, L. Schares, G. Guekos, S. Hansmann, and R. Dall'Ara, "20 Gb/s all-optical XOR with UNI gate," *IEEE Photon. Technol. Lett.*, vol. 12, pp. 834–836, July 2000.

[25] K. E. Stubkjaer, "Semiconductor optical amplifier-based all-optical gates for high-speed optical processing," *J. Select. Topics Quantum Electron.*, vol. 6, no. 6, pp. 1428–1435, 2000.

[26] Y. Oie, T. Suda, M. Murata, D. Kolson, and H. Miyahara, "Survey of switching techniques in high speed networks and their performance," *Proc. IEEE INFOCOM 90*, pp. 1242–1251, 1990.

[27] M. A. Henrion, G. J. Eilenberger, G. H. Petit, and P. H. Parmentier, "A multipath self-routing switch," *IEEE Commun. Mag.*, vol. 31, pp. 46–52, Apr. 1993.

[28] S. P. Monacos, J. M. Morookian, L. Davis, L. A. Bergman, S. Forouhar, and J. R. Sauer, "All-optical WDM packet networks," *J. Lightwave Technol.*, vol. 14, pp. 1356–1370, June 1996.

[29] S. Banerjee, V. Jain, and S. Shah, "Regular multihop logical topologies for lightwave networks," *IEEE Commun. Surv.*, 1999.

[30] S. Banerjee, J. Yoo, and C. Chen, "Design of wavelength-routed optical networks for packet switched traffic," *J. Lightwave Technol.*, vol. 15, no. 9, 1997.

[31] C. K. Ko and S. Y. Kuo, "Multiaccess processor interconnection using subcarrier and wavelength division multiplexing," *J. Lightwave Technol.*, vol. 15, pp. 228–241, Feb. 1997.

[32] O. Komolafe, D. Harle, and D. Cotter, "Impact of graph theoretic network parameters on the design of regular virtual topologies for optical packet switching," *Proc. IEEE ICC*, vol. 5, pp. 2827–2831, 2002.

[33] C. Y. Li, P. K. A. Wai, X. C. Yuan, and V. O. K. Li, "Deflection routing in slotted self-routing networks with arbitrary topology," *Proc. IEEE ICC*, pp. 2781–2785, 2002.

[34] A. J. Poustie, K. J. Blow, A. E. Kelly, and R. J. Manning, "All-optical parity checker," in *Proc. Optical Fiber Communication Conf.; Int. Conf. Integrated Optics and Optical Fiber Commun.*, vol. 1, 1999, pp. 137–139.

[35] W. Hung, K. Chan, L. K. Chen, C. K. Chan, and F. Tong, "A routing loop control scheme in optical layer for optical packet networks," in *Proc. Optical Fiber Commun. Conf. Exhibit* , 2002, pp. 770–771.

[36] W. J. Goralski, *Inroduction to ATM Networking*. New York: MacGraw–Hill, 1995, ch. 6.

[37] K. Kato, "Ultrawide-band/high-frequency photodetectors," *IEEE Trans. Microwave Theory Tech.*, vol. 47, pp. 1265–1281, July 1999.

[38] H. M. Ji and R. Srinivasan, "Fast IP routing lookup with configurable processor and compressed routing table," *Proc. IEEE GLOBECOM 01*, pp. 2373–2737, 2001.

[39] E. L. Wooden *et al.*, "A review of lithium niobate modulators for fiber-optic communication systems," *IEEE J. Select. Topics Quantum Electron.*, vol. 6, pp. 69–82, Jan. 2000.

[40] The network topology is obtained from the URL (2002, Apr.). [Online]. Available: http://www.ipservices.att.com/backbone/bbone-map.cfm

[41] The network topology is obtained from the URL (2002, Apr.). [Online]. Available: http://www.dante.net/geant/Schematic.jpg

[42] N. J. Doran and D. Wood, "Nonlinear-optical loop mirror," *Opt. Lett.*, vol. 13, pp. 56–58, 1988.

[43] N. A. Whitaker Jr, H. Avramopoulos, P. M. W. French, M. C. Gabriel, R. E. LaMarche, D. J. DiGiovanni, and H. M. Presby, "All-optical arbitrary demultiplexing at 2.5 Gb/s with tolerance to timing jitter," *Opt. Lett.*, vol. 16, pp. 1838–1840, 1991.

[44] M. Jinno and T. Matsumoto, "Ultrafast all-optical logic operations in a nonlinear Sagnac interferometer with two control beams," *Opt. Lett.*, vol. 16, pp. 220–222, 1991.

[45] K. L. Hall and K. A. Rauschenbach, "All-optical bit pattern generation and matching," *Electron. Lett.*, vol. 32, pp. 1214–1215, 1996.

[46] K. Uchiyama, H. Takara, T. Morioka, S. Kawanishi, and M. Saruwatari, "100 Gbit/s multiple-channel output all-optical demultiplexing based on TDM-WDM conversion in a nonlinear optical loop mirror," *Electron. Lett.*, vol. 32, pp. 1989–1991, 1996.

[47] M. Eiselt, W. Pieper, and H. G. Weber, "SLALOM: Semiconductor laser amplifier-based processing nodes," *J. Lightwave Technol.*, vol. 13, pp. 2099–2112, 1995.

[48] A. D. Ellis, D. M. Patrick, D. Flannery, R. J. Manning, D. A. O. Davies, and D. M. Spirit, "Ultra-high speed OTDM networks using semiconductor amplifier-based processing nodes," *J. Lightwave Technol.*, vol. 13, pp. 761–770, 1995.

[49] T. J. Xia, Y. Liang, K. H. Ahn, J. W. Lou, O. Boyraz, Y. H. Kao, X. D. Cao, S. Chaikamnerd, J. K. Andersen, and M. N. Islam, "All-optical packet-drop demonstration using 100-Gb/s words by integrating fiber-based components," *IEEE Photon. Technol. Lett.*, vol. 10, pp. 153–155, Jan. 1998.

[50] P. Toliver, I. Glesk, and P. R. Prucnal, "Comparison of three nonlinear interferometer optical switch geometries," *Opt. Commun.*, vol. 175, pp. 365–373, 2000.

[51] R. Ramaswami and K. N. Sivarajan, *Optical Networks: A Practical Perspective*. New York: Morgan-Kaufmann, 1998, ch. 14.

[52] E. W. Zegura, K. I. Calvert, and S. Bhattacharjee, "How to model an internetwork," *Proc. IEEE INFOCOM 96*, vol. 2, pp. 594–602, 1996.

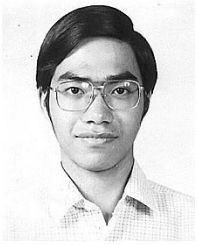
X. C. Yuan was born in China in 1974. He received the B.S. degree in computer science from Wuhan University, Wuhan, China, in 1993 (having been admitted at age 15). He received the M.S. degree from the Institute of Computing of the Chinese Academy of Science, Beijing, China, in 1996 and the Ph.D. degree in electrical and electronic engineering from the University of Hong Kong, Hong Kong, in 2002.

From 1996 to 1998, he worked at the Chinese Academy of Science. In 1998, he joined the University of Hong Kong as a Research Assistant. His research area is in optical network routing, and he has published five papers in this area. He has also published three books on the Internet, JAVA programming, and website design.

Victor O. K. Li (S'80–M'81–SM'86–F'92) was born in Hong Kong in 1954. He received the S.B., S.M., E.E., and Sc.D. degrees in electrical engineering and computer science from the Massachusetts Institute of Technology (MIT), Cambridge, in 1977, 1979, 1980, and 1981, respectively.

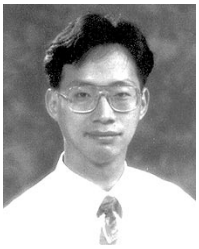
He joined the University of Southern California (USC), Los Angeles, in February 1981 and became Professor of Electrical Engineering and Director of the USC Communication Sciences Institute. Since September 1997, he has been with the University of Hong Kong, Hong Kong, where he is Chair Professor of Information Engineering at the Department of Electrical and Electronic Engineering, as well as Managing Director of Versitech, Ltd. (http://www.versitech.com.hk/), Hong Kong, the technology transfer and commercial arm of the University. He also serves on various corporate boards. He is a Principal Investigator of the Area of Excellence in Information Technology funded by the Hong Kong government. Sought by government, industry, and academic organizations, he has lectured and consulted extensively around the world. His research interest is in information technology, including high-speed communication networks, wireless networks, and Internet technologies and applications.

Prof. Li chaired the Computer Communications Technical Committee of the IEEE Communications Society (1987–1989) and the Los Angeles Chapter of the IEEE Information Theory Group (1983–1985). He cofounded the International Conference on Computer Communications and Networks (IC3N) and chaired its Steering Committee (1992–1997). He also chaired various international workshops and conferences and was most recently appointed General Chair of IEEE INFOCOM 2004. He has served as an Editor of IEEE NETWORK, IEEE JOURNAL OF SELECTED AREAS IN COMMUNICATIONS (JSAC) Wireless Communications Series, and Telecommunication Systems. He was also a Guest Editor for special issues of IEEE JSAC, *Computer Networks and ISDN Systems*, and KICS/IEEE JOURNAL OF COMMUNICATIONS AND NETWORKING. He is now serving as an Editor of ACM/Kluwer *Wireless Networks* and IEEE Communications Surveys and Tutorials. He was recently appointed to the Hong Kong Information Infrastructure Advisory Committee by the Chief Executive of the Hong Kong Special Administrative Region. He also serves on the Innovation and Technology Fund (Electronics) Vetting Committee, the Small Entrepreneur Research Assistance Programme Committee, the Engineering Panel of the Research Grants Council, and the Task Force for the Hong Kong Academic and Research Network (HARNET) Development Fund of the University Grants Committee. He was a Distinguished Lecturer at the University of California at San Diego, at the National Science Council of Taiwan, and at the California Polytechnic Institute. He has also delivered keynote speeches at many international conferences. He has received numerous awards, including, most recently, the Croucher Foundation Senior Research Fellowship in March 2002 and the Bronze Bauhinia Star, Government of the Hong Kong Special Administrative Region, China, July 1, 2002.



C. Y. Li (S'91–M'93) received the B.S. degree from the National Taiwan University, Taipei, Taiwan, in 1986, and the Ph.D. degree from the Hong Kong Polytechnic University, Kowbon, Hong Kong, in 2000.

In 1986, he joined Taicom, Ltd., where he worked as a Transmission Engineer on the M90, M135, and M405 Optical Fiber Telecommunication System projects. In 1988, he joined ROCTEC, Ltd., where he worked as a Design Engineer on the computer products development. In 1993, he joined the Department of Electronic and Information Engineering, Hong Kong Polytechnic University, as a Research Assistant, and then became a Ph.D. candidate. After receiving the Ph.D. degree, he worked as a Research Associate in the same department. His research interests include network performance evaluation, all-optical network routing, and network theory, and he has published 13 papers in these areas.



P. K. A. Wai (SM'96) received the B.S. degree with first-class honors from the University of Hong Kong in 1981 and the M.S. and Ph.D. degrees from the University of Maryland, College Park, in 1985 and 1988, respectively.

In 1988, he joined Science Applications International Corporation in McLean, VA, where he worked as a Research Scientist on the Tethered Satellite System project. In 1990, he became a Research Associate in the Department of Electrical Engineering, University of Maryland. In 1996, he joined the Department of Electronic and Information Engineering, Hong Kong Polytechnic University, Kowbon, Hong Kong, as an Assistant Professor. He became Associate Professor in 1997 and Professor and Head of the Department in 2002. His research interests include the theory of solitons, the modeling of fiber lasers, simulations of integrated optical devices, long-distance optical communications, all-optical packet switching, and network theories. He is an active contributor to the technical field, having contributed to more than 100 international publications.

Prof. Wai is a Member of Optical Society of America (OSA).