A novel self-routing scheme for all-optical packet switched networks with arbitrary topology

X. C. Yuan, Victor O. K. Li,
Department of Electrical and Electronic Engineering
The University of Hong Kong, Hong Kong
{xcyuan, vli}@eee.hku.hk

C. Y. Li, and P. K. A. Wai
Department of Electronic and Information Engineering
The Hong Kong Polytechnic University, Hong Kong
{enli, enwai}@polyu.edu.hk

Abstract—**Due to limited available photonic devices, optical networks in the near future will likely employ routing schemes that do not require sophisticated processing of optical packets. In this paper, we propose a novel self-routing scheme for all-optical packet networks that can be applied to networks with arbitrary topology. The proposed routing scheme requires only single bit processing and can be implemented with existing technologies.**

I. INTRODUCTION

To avoid the electronics bottleneck, all-optical communication networks are likely to be the networks of the future [1], [2]. Terabit networks utilizing wavelength division multiplexing (WDM) technology have already been demonstrated [3]. Most existing WDM networks employ circuit switching technology, and thus are not suitable for data applications involving bursty traffic. Implementation of optical packet switched networks is difficult because of the lack of practical optical buffers and limited photonic devices available when compared to electronics [1]. At present, fiber delay lines are used as First In First Out (FIFO) optical buffers. As optical buffers are far from practical, many bufferless all-optical packet switched networks are proposed. They include the use of hot-potato routing to re-route the contention packets [4], and retransmission of the contention packets in different wavelengths [5]. The former method increases latency while the latter requires wavelength converters. Both approaches require relatively complex optical routing control. Of course, one may set up dedicated lightpaths for packet transmission, such as in Lambda-labeling [6]. However, the overhead in lightpath setup is typically much larger than the duration of a packet.

The optical logic device technology is still in its infancy when compared to electronics. Complex optical logic circuits are not feasible yet. One solution for optical packet networks is to transmit the packet header at a lower speed channel, *e.g.,* optical subcarrier multiplexing [7], optical burst switching [8], or transmission of the packet header at the same channel but with a lower bit rate [9]. However, these methods are sensitive to the synchronization between the processing of the packet payload and the packet header.

Self-routing schemes are attractive for optical packet switched networks because they simplify routing control and require single bit processing only. Intermediate nodes forward the incoming packets to the appropriate output ports using bit-by-bit comparison of the packet headers. No table lookup function is required. The packet routing speed is limited by the hardware speed only. Traditionally, self-routing schemes can only be applied to networks with regular topology such as hypercube, ShuffleNet, and Manhattan Street networks. To apply self-routing to networks with arbitrary topology, it is necessary to map the physical topology of the networks to logical networks with regular topology. Such mapping cannot be carried out in general. Besides, all paths between nodes are fixed in self-routing schemes. It is difficult to implement congestion control and traffic engineering. Rerouting of the paths for system reconfiguration is also difficult. In this paper, we propose a novel self-routing scheme applicable to networks with arbitrary topology. Only single bit processing is required for packet routing. The proposed scheme allows multiple addresses for the same node. Each address encodes a different set of paths from other nodes to this node. Multiple paths between two nodes are therefore possible. The proposed self-routing scheme can be readily adapted to a hierarchical structure for use in hierarchical networks.

II. THE SELF-ROUTING SCHEME

In the proposed self-routing scheme, the paths between any two nodes are fixed. The address of a node encodes a unique path from any other node to the node itself. If there is more than one path, 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 contained in each address must satisfy 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.*

If a node has multiple addresses, the paths encoded in each address must satisfy the condition above. The routing information encoded in each address, for the same node or different nodes, are independent of one another.

This research is supported in part by the University Grants Committee, Hong Kong, Area of Excellence in Information Technology, Grant No. AOE98/99.EG01. Additional support is provided by a grant from The Hong Kong Polytechnic University (Project No. A-PB97).

0-7803-7097-1/01/$10.00 ©2001 IEEE

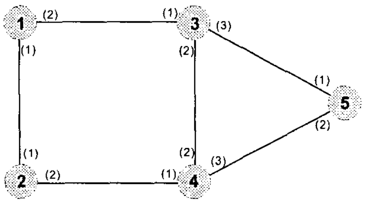

Fig. 1. A 5-node 6-link network.

TABLE I
THE 20 PATHS BETWEEN THE NODES FOR THE NETWORK IN FIG. 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

A. Address structure

We consider a network made up of N nodes and L links. For simplicity, all links are assumed to be bi-directional. The proposed scheme can be applied to uni-directional links as well. Each node is arbitrarily labeled from 1 to N. The links connecting to each node is also arbitrarily labeled from 1 to $n(i)$ where $n(i)$ is the number of links connected to the i-th node. We have $\sum_{i=1}^{N} n(i) = 2L$.

The address of a node contains H bits, where $H = 2L$. Each address is divided into N fields. Each field corresponds to one node in the network. The i-th field of an address contains $n(i)$ bits. Each address field contains the instruction to handle the packet at the node corresponding to the address field. The i-th address field of node i is set to zero. For the j-th address field of node i, $j \neq i$, one and only one of the $n(j)$ bits, the x-th bit say, is set to **1**. The other bits at the j-th address field is set to zero. A non-zero entry at the x-th bit of the j-th address field means that node j will forward a packet with such an address to the x-th output link.

When a node receives a packet, it only processes the address field corresponding to the node itself. A node recognizes that a packet has arrived at the destination if the corresponding address field is all zeroes; otherwise, it forwards the packet to the local output link as specified. There is a total of $(N-1)$ **1** bits out of the H bits in each address. Bits in an address field are set to **1** depending on the paths defined.

As an illustration, we consider the 5-node 6-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 a sequence of nodes. 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 links of each node with numbers in parentheses as shown in Fig. 1. The labeling of the link at each node has local significance only. There are 5 fields in the node address corresponding to the 5 nodes in the network. The number of bits in each address field, $n(i), i = 1, \ldots, 5$, is given by $n(1) = 2$, $n(2) = 2$, $n(3) = 3$, $n(4) = 3$, and $n(5) = 2$. The total number of bits $H = 12$.

The address of node 1 is constructed as follows. The bits in the first field are set to zero, *i.e.*, the first two bits of the address is **00**. For the second field, we look at the connection from node 2 to node 1 which is the path $P(2,1) = 21$. Since a packet sent from node 2 to node 1 is transmitted through the link 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 of the address of node 1 is therefore **10.** Similarly for the third address field, we look at the connection from node 3 to node 1. The third address field of the address of node 1 is therefore **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 link (1), and then from node 2 to node 1 through link (1). From the first part of the routing instruction, the fourth address field is given by **100.** The packet is now at node 2. So we look at the second address field of the address. From the second part of the routing instruction, the second address 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. Similarly, we check path $P(5,1)$ to determine the fifth address field. We find that path $P(5,1) = 5431$ is in conflict with the contents of the first 4 address fields because the four paths violate Condition 1. In the path $P(5,1)$, after a packet is routed to node 4, it is then sent to node 3 instead of node 2 as is the case in $P(4,1)$. To accommodate this path, 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 actually be 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.

In general, the set of paths to a node do not comply with Condition 1. If a unique address for the node is desired, it is necessary to modify some of the paths. If the paths cannot be changed, one can add new paths to the node such that multiple addresses of the node can be formed. The union of all the paths encoded in the multiple addresses should contain all the original paths. The additional paths can be made fictitious if we restrict the use of the addresses in some of the nodes.

0-7803-7097-1/01/$10.00 ©2001 IEEE

TABLE II
THE SELF-ROUTING NODE ADDRESSES FOR THE NETWORK IN FIG. 1 AND THE PATHS SHOWN IN TABLE I.

Field: Address	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

The self-routing addresses of the five nodes of the network corresponding to the routing paths shown in Table I is given in Table II. 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. Note that one can replace address 1a of node 1 by the address **00_10_10_010_10** without affecting the addresses of other nodes. In this case, a packet sent from node 5 to node 1 will go to node 3 directly without a detour to node 4. This demonstrates that the routing information contained in different addresses is independent of one another.

B. Routing of packets

When all the node addresses are properly assigned, a packet can be automatically routed to its destination following the predefined path. Let us consider the transmission of a packet from node 1 to node 5. The user data is encapsulated in a packet having the address of node 5 as the destination address and is sent to node 1. We assume that address 5a is chosen. Node 1 checks the first address field bit by bit. As the second bit in the first address field is set to **1**, node 1 routes the packet to the node connected to the second output port, *i.e.*, node 3. When node 3 receives this packet, it checks the third address field bit by bit. As the third bit of the address field is set to **1**, node 3 routes the packet to the node connected to the third output port, *i.e.*, node 5. When node 5 receives the packet, it retrieves the user data from the packet because no bit in the fifth address field is set to **1**. The routing path of the packet is 135.

III. OPTICAL IMPLEMENTATIONS AND PACKET SWITCHING TIME

In recent years, there has been intense research in optical signal processing. Logic OR and NOR have been demonstrated at 10 Gb/s using an ultrafast nonlinear interferometer. The XOR functionality has been demonstrated using a terahertz optical asymmetric demultiplexer (TOAD) and also integrated semiconductor optical amplifier (SOA) based Mach-Zehnder interferometer (MZI). SOA-based interferometric demultiplexer can also be used to perform logic OR. In [10], an optical packet-switching testbed capable of performing message routing at 100 Gb/s is demonstrated. The testbed uses a structured and highly interconnected network topology known as the ShuffleNet such that a self-routing protocol can be developed to minimize the complexity of routing control. In the two-connected 8-node ShuffleNet studied in [10], each node is connected to two output ports. The routing instruction is encoded in four 2-bit groups. The bit patterns **01**, **10**, and **11** represent the "route down," "route up," and "receive packet" action that can be taken by the current node. With only slight modifications, the proposed self-routing scheme can be implemented using the technologies demonstrated in [10].

The packet switching time depends on the header processing time and the packet routing approaches used. When a node receives a packet, it only checks the bit field corresponding to this node. Depending on the positions of these bits in the header, the delay can be as long as the duration of the header. The header may be several kilobits long for wide area networks with thousands of nodes. The overhead however is often negligible in practice when compared to the payload of the optical packets. The payload of a typical optical packet in general will be several hundred kilobytes long because an optical packet will contain thousands of Internet Protocol (IP) packets [6]. Thus the header overhead of the proposed self-routing scheme will only be 1 to 2% of the total packet length which is smaller than that of most protocols. For comparison, the header overhead of Asynchronous Transfer Mode is about 9%.

Currently there are two packet routing approaches: all-optical and hybrid. In an all-optical approach, both the data and the control signals remain in the optical domain from source to destination. All-optical routing using TOAD as an all-optical routing switch has been demonstrated [11]. The routing delay is on the order of the bit period. In a hybrid approach, the data signal remains in the optical domain, but the header is split off and converted into the electrical domain at each node and fed into an electronic routing controller. The routing controller then sets the state of an electro-optic routing switch such as a $LiNbO_3$ crossbar switch. Picosecond switching time $LibNbO_3$ electro-optic switch with small number of ports, *e.g.*, 2×2, is commercially available. Larger size electro-optic switches with nanosecond switching time have also been reported [2]. Consequently, the overall switching delay using the proposed self-routing scheme with existing technologies will be a small fraction of the packet duration. For comparison, the duration of a 100 kilobyte long packet will be 80 microseconds at 100 Gb/s. The end to end propagation delay between two nodes a hundred kilometers apart is in miniseconds.

IV. 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 a path from a source node to this node fails, the source node can switch to a different path by using the other address. The 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 the new routing information. 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 are required 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 interrupt services. To minimize the disruptions, additional bits can be reserved in the header for assigning 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 between nodes 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.

V. Extensions

A. Address compression

We observe that in many occasions some of the bits 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 the 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.

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 address bits by changing some of these bits from zeroes to ones. Furthermore, if two bits complement each other in all the addresses, then one of the bits can be removed provided that the complement of a bit value in an address field is available.

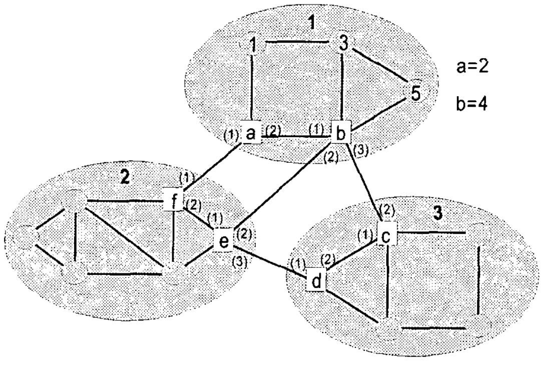

Fig. 2. A network with 3 subnetworks.

B. Hierarchical address

Hierarchical addressing can be used with the proposed self-routing scheme in order to take advantage of the hierarchical nature of the networks. In principle, there is no limit to the number of hierarchical levels in the address. In practice, one does not expect an address structure with more than three or four levels. In the following, we discuss the address structure of a two-level network. One can easily extend this to three or more levels. An example of a two-level network is shown in Fig. 2. The network is divided into three subnetworks interconnected through some of the nodes. From the types of connections, one can define two types of nodes; subnetwork nodes and border nodes. Subnetwork nodes are connected to nodes belonging to the same subnetwork only. Border nodes of a subnetwork are connected to the border nodes of other subnetworks as well as to nodes of the same subnetwork. A subnetwork may have more than one border node. In Fig. 2, the nodes labeled a, ..., f are border nodes. The nodes labeled 1, 3, 5 are subnetwork nodes of subnetwork 1. The nodes a and b are the border nodes of subnetwork 1. The subnetwork nodes form the intra-subnetwork layer of the hierarchical network and the border nodes form the inter-subnetwork layer of the hierarchy.

The hierarchical address is made up of three parts corresponding to the three parts of the routing: (1) the routing of a packet from the source node to a border node of the source subnetwork, (2) the routing of the packet from the border node of the source subnetwork to a border node of the destination subnetwork, and (3) the routing of the packet from the border node of the destination subnetwork to the destination node. The first part contains the intra-subnetwork address of a node. The intra-subnetwork address is encoded in exactly the same way as described in Section II. For the inter-subnetwork routing, *i.e.*, among the border nodes, the addresses are constructed in a similar fashion except that the address fields of all the border nodes of the subnetwork to which the nodes belong are set to zero. In order to avoid ambiguity in routing instructions, once a packet arrives at the first border node of the destination subnetwork, the rest of the route is considered intra-subnetwork and is handled by the intra-subnetwork address of the destination node. Since the border nodes of a subnetwork participate in both inter-subnetwork and intra-subnetwork routing, they have address fields in both the inter-subnetwork and intra-subnetwork part of the address.

The routing from the source node to the border nodes of the source subnetwork is encoded in the third part of the address. This part of the address is constructed as follows. The number of bits is equal to the number of intra-subnetwork addresses of all the border nodes. Recall that a node can have multiple addresses. Each bit is associated with one intra-subnetwork address of the border nodes. The bits corresponding to the border nodes of the same subnetwork form an address field of the subnetwork. At each subnetwork node, an output port is assigned to each bit position in the address field corresponding to the subnetwork. The assignment is in accordance with the routing instruction encoded in the intra-subnetwork address of the border node associated to that bit. Consequently, if a bit in the address field of a subnetwork is set to one, a packet at any node inside the subnetwork is routed to the corresponding border node using the paths encoded in the associated intra-subnetwork address. Only one bit in the address field of a subnetwork is set to one.

In order to properly route a packet, a node must be able to determine whether a packet belongs to the subnetwork of the node. The routing of a packet is performed in the following way. When a subnetwork node receives a packet, it first checks the inter-subnetwork part of the address. Specifically, it checks the address fields corresponding to any one of the border nodes of the subnetwork. It is only necessary to check one of the border node address fields because by construction, the fields either all contain zeroes or all contain one non-zero element. If the address field is zero, the packet belongs to the subnetwork. The node then checks its own address field in the intra-subnetwork part of the address and routes the packet accordingly. If the address field of the border nodes contains a non-zero element, the packet is destined for another subnetwork and should be routed to the border node. The node then inspects the third part of the address. It checks the address field corresponding to its own subnetwork and routes the packet to the output port according to which bit is set in this field.

When a border node receives a packet, it checks its own address field in the inter-subnetwork part of the address. If it is zero, the packet belongs to the subnetwork of the border node. It then checks its own address field in the intra-subnetwork part of the address and routes the packet accordingly. If the address field in the inter-subnetwork part of the address contains a non-zero element, the packet does not belong to the subnetwork and is routed according to the instruction encoded in this field.

The hierarchical address also requires single bit processing only and can be implemented with the technologies demonstrated in [10]

VI. Conclusion

In this paper, we propose a self routing scheme for all optical packet switched networks. Compared to traditional self routing schemes, the proposed method can be implemented with any network topology. There is no restriction on the 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 alternative routing in case of network failure and for congestion control. The proposed scheme requires only single bit optical processing and can be implemented with current technology. With the use of extra unassigned address bits, we can reduce the chance of network-wide reconfiguration when nodes and links are added. We have also demonstrated that the proposed scheme can be combined with hierarchical addressing to reduce the length of the address.

References

[1] V.W.S. Chan, K.L. Hall, E. Modiano, and K.A. Rauschenbach, "Architectures and technologies for high-speed optical data networks," *Journal of Lightwave Technology*, Vol. 16, No. 12, pp. 2146-2168, 1998.

[2] R. Ramaswami and K.N. Sivarajan, *Optical networks: A practical perspective*, Morgan Kaufmann Publishers, Inc., San Francisco, USA, 1998.

[3] S. Aisawa, T. Sakamoto, M. Fukui, J. Kani, M. Jinno, and K. Oguchi, "Ultra-wideband, long distance WDM demonstration of 1 Tbit/s (50 × 20Gbit/s), 600km transmission using 1550 and 1580nm wavelength bands," *IEE Electronics Letters*, Vol. 34, No. 11, pp. 1127-1128, 1998.

[4] A.S. Acampora and S.I.A. Shah, "Multihop lightwave networks: a comparison of store-and-forward and hot-potato routing.," *IEEE Transactions on Communications*, Vol. 40, No. 6, pp. 1082-1090, 1992.

[5] K.C. Lee and V.O.K. Li,"Optimization of a WDM optical packet switch with wavelength converters," *IEEE Infocom 95*, pp. 423-430, 1995.

[6] N. Ghani, *et al.*, "Lambda-labeling: A framework for IP-over-WDM using MPLS," *SPIE Optical Networks Magazine*, Vol. 1, No. 2, pp. 45-58, 2000

[7] A. Carena, M.D. Vaughn, R. Gaudino, M. Shell, and D.J. Blumenthal, "OPERA: an optical packet experimental routing architecture with label swapping capability," *Journal of Lightwave Technology*, Vol. 16, No. 12, pp. 2135-2145, 1998.

[8] C. Qiao and M. Yoo, "Optical burst switching (OBS) - a new paradigm for an optical Internet," *Journal of High Speed Networks*, Vol. 8, pp. 69-84, 1999.

[9] M. Renaud, C. Janz, P. Gambini, and C. Guillemot, "Transparent optical packet switching: The European ACTS KEOPS project approach," *IEEE Lasers and Electro-Optics Society, LEOS'99*, Vol. 2, pp. 401-402, 1999.

[10] P. Toliver, I. Glesk, R.J. Runser, Kung-Li Deng, B.Y. Yu, and P.R. Prucnal, "Routing of 100 Gb/s words in a packet-switched optical networking demonstration (POND) node," *Journal of Lightwave Technology*, Vol. 16, No. 12, pp. 2169-2180, 1998.

[11] I. Glesk, J.P. Solokoff, and P.R. Prucnal, "All-optical address recognition and self-routing in a 250 Gbit/s packet switched network," *IEE Electronics Letters*, Vol. 30, No. 16, pp. 1322-1323, 1994.

0-7803-7097-1/01/$10.00 ©2001 IEEE