Synchronous Flow Control in Wormhole Routed Optical Networks

Chang-Ching Sue and Sy-Yen Kuo

Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan
Email: sykuo@cc.ee.ntu.edu.tw
URL: lion.ee.ntu.edu.tw

Abstract

In this paper [1], we propose a synchronous flow control mechanism in wormhole routed optical networks. It is expected that the benefit of shorter routing delay and smaller buffer size requirement in wormhole routing will be significant in optical networks. Different from the traditional bi-directional asynchronous backpressure flow control, the flow control is modified to be unidirectional and synchronous. The size of synchronized control slot does not depend on the routing path length and the number of bits is a constant which is equal to the total number of virtual channels and nodes. The proposed flow control takes advantage of the restricted order of accessing channels in deadlock-free routing to broadcast their control information in a corresponding restricted order . Furthermore, in order to reduce the buffer size to only one unit, the virtual channels which share the same physical channel must be able to simultaneously transmit data. The low channel utilization induced by such mechanism is overcome by our modified source routing. In summary, this paper introduces a flow control mechanism which easily incorporates the benefit of wormhole routing into the limited-resource optical networks.

1. Introduction

Optical networks, which have the potential to provide much higher throughput than electronic networks, are extensively implemented with Wavelength Division Multiplexing (WDM) technologies [4,5]. Wormhole routing, which was proposed to reduce the routing delay and the buffer requirement in the conventional electronic networks, can also be applied to WDM networks with appropriate modifications.

Wormhole routing has been implemented on optical LANs. The ARPA-sponsored Supercomputer Super Net (SSN) project at UCLA [1,2], which is a two-level hierarchical campus-wide network evolved from the research of Myrinet [3], extended the wormhole routing, source routing and backpressure flow control of Myrinet in a transparent fashion. Myrinet is based on such simple network protocols and therefore, is prone to congestion and deadlock. These are crucial problems in very high speed networks, where large amount of information can be lost when network resources become unavailable. Some form of flow control must thus be introduced to prevent congestion. A flow control scheme in Myrinet is backpressure. Backpressure is an explicit link-by-link flow control mechanism requiring bidirectional links. Due to extension in a transparent manner, the protocols used in SSN are identical to those employed in Myrinet. Therefore, the optical links in SSN are bidirectional in order to facilitate the backpressure hop-by-hop flow control [8]. However, directly replacing the bit-parallel copper cables with the WDM channels in fiber cables does not produce bidirectional communication. The bidirectional communication in optical networks requires double wavelengths and transceivers. Given that the number of wavelengths in an optical fiber is limited, unidirectional flow control in wormhole routing networks should be more appropriate for optical networks.

Although the wormhole routing on the multihop and bi-directional optical networks is properly implemented by SSN-associated studies, a uni-directional flow control, which uses only half of the fixed optical devices, is not yet studied. In this paper, we assume that loss

[1] Acknowledgement: This research was supported by the National Science Council, Taiwan, R. O. C., under the Grant NSC 87-2213-E259-007

of worms inside the network must be avoided. Hence, deadlock avoidance must be employed. A deadlock-free wormhole routing flow control mechanism in uni-directional multi-hop optical star networks is proposed. Instead of deflection routing [14] and restricted routing [15], deadlock-free is guaranteed by an approach called virtual channels, combine restricted routing and buffer partition to achieve deadlock free routing in the wormhole routing context [9]. In our modified virtual channel flow control, virtual channels on a single physical channel can be served simultaneously. This makes the buffer size equal to one and simplifies the design of controller.

The low utilization of one data slot is overcome by modified source routing, which takes advantage of the concept of virtual channels. In our modified source routing, the path generated by the source node is not always from the lowest channel to the highest channel but from the lower to the higher. For hardware requirement, one external virtual channel status table, one local virtual channel status table , and one transmitter status table in the controller of each node are the additional control components in our mechanism. Furthermore, a few modifications are made on the flit's format. We take Shufflenet [10-12]as an example for explanation of our mechanism. The same method could be applied to regular multi-hop networks. The results of this study may be of interest to those who attempt to combine the throughput advantage of optical networks with the low latency benefit and the smaller buffer size advantage of wormhole routing technique in a cost-effective manner.

This paper is organized as follows. In the second section we provide an overview of deadlock-free routing and introduce our channel numbering principle. The third section presents the architecture and operation for synchronous unidirectional flow control in optical networks according to the channel number obtained in Section two. The fourth section presents a simulation model which will be used to compare the different control schemes and simulation results are presented. The final section concludes the paper.

2. Overview of wormhole routing

The authors in [9] introduced a routing technique called wormhole routing, which overcomes the long delay disadvantage in store-and-forward routing techniques. A disadvantage of wormhole routing is that deadlocks may occur if cyclic waiting is created by the routing algorithm. The paper [9] presented deadlock-free routing algorithms for several network topologies such as the k-ary n-cube, the Shuffle-Exchange network and the Cube-connected Cycles. In this paper, we present a deadlock-free algorithm similar to [7] for the Shufflenet topology.

The following subsection introduces the numbering principle of virtual channels for deadlock-free wormhole routing in shufflenet. Then the next section also takes shufflenet as an example for explaining how synchronous uni-directional flow control is implemented.

2.1. Deadlock-free wormhole routing for shufflenet

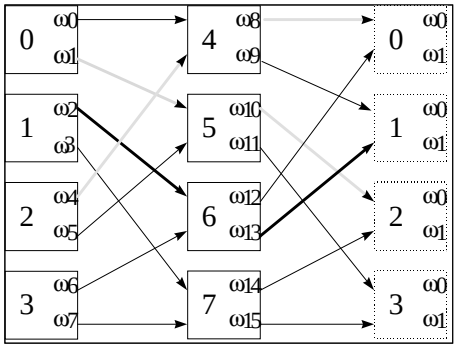

Figure 1. An 8-node (2,2) shufflenet.

A Shufflenet with parameters p and k has $N = kp^k$ nodes. It has k columns of p^k nodes each. Each node has p transmitters and p receivers. The shufflenet multihop topology has several cycles of varying lengths. For instance, in the eight node shufflenet shown in Fig. 1, node 1→node 6→node 1 and node 0→node 5→node 2→node 4→node 0 are cycles. Thus, if special precautions are not taken to avoid cycles, deadlocks could happen. In the following, we first define some notations and then present the routing function R followed by a proof of its correctness.

There are many possible routing algorithms for Shufflenet, including fixed and adaptive routings. Although the adaptive routing schemes could be better than the fixed routing ones in the existence of non-uniform traffic patterns, this is not our major cause for concern. There are more complicated fixed routing schemes that alleviate the load imbalances. In this section, we present one that is particularly simple to implement and will serve to demonstrate the deadlock-free property of routing algorithms.

We define the following notations. The k columns are numbered 0 to k-1 and a wrap around occurs when a node in column i sends a message to a node in column j, where $i \geq j$. The p^k nodes in a single column are numbered from 0 to p^k-1. Thus each node i of the N nodes in the shufflenet can be uniquely identified by

(c,r) where $c \in [0, k-1]$, $r \in [0, p^k - 1]$ and $i = c \cdot p^k + r$. Note that the row coordinate r is represented in p-ary notation. That is,

$$r = r_{k-1} r_{k-2} ... r_0$$

where the r_i's are the base p digits in the representation, and

$$r = \sum_{i=0}^{k-1} r^i \cdot p^i$$

.

According to the Shufflenet's connectivity, the following p nodes are one hop away from (c,r):

$((c+1) mod k, r_{k-2} r_{k-3} ... r_0 0)$,

$((c+1) mod k, r_{k-2} r_{k-3} ... r_0 1)$,

..

$((c+1) mod k, r_{k-2} r_{k-3} ... r_0 (p-1))$.

The total number of physical channels in a shufflenet is kp^{k+1}. We number the channels from 0 to kp^{k+1}-1 and denote them by C_j where j is the channel number (see Fig. 1). Now, we map 3 virtual channels to each physical channel in the following fashion. The 3 virtual channels associated with a physical channel C_j are denoted as C_j, $C_{kp^{k+1}+j}$, $C_{2 \cdot kp^{k+1}+j}$, where $j \in [0, kp^{k+1} - 1]$. The queue associated with each physical channel at a node's input port is also divided into 3 parts - one for each virtual channel with the same id as the virtual channel's subscripts. The source-generated message is placed in the buffer associated with the source node. The source buffer is numbered as a negative number, -C_n. Source routing is adopted, so decisions of the routing path are made at the source node and not distributed in other nodes in the routing path. The routing path is generated according to the following function.

When a node S with id (cs,rs) generates a message to be sent to another node D with id (cd,rd), the routing decision of the routing path at the source node is based on the p-ary representation of source address (cs,rs) and destination address (cd,rd).

If we let D denote the number of columns between the source (cs, rs) and the destination (cd,rd), then

$$D = \begin{cases} cd - cs & if \quad cd > cs \\ k + cd - cs & if \quad cd \leq cs \end{cases}$$

Since the column address increases by one after each hop, D represents a lower bound on the number of hops required to route a message from (cs,rs) to (cd,rd). However, it may not be possible to reach the destination in D hops. For example, in Fig. 1 it takes three hops to route a message from node 1 to node 5, even though they are only one column apart.

The scheme attempts to deliver the packet in D hops by routing successively according to the last D p-ary digits of rd. In other words, from node

$S = (cs, rs) = (cs, r_{k-1s} r_{k-2s} .... r_{0s})$

the flit is first routed to the node with address

$(cs1, rs1) = ((cs+1) mod k, r_{k-2s} r_{k-3s} ... r_{0s} r_{D-1d})$,

and then to

$(cs2, rs2) = ((cs+2) mod k, r_{k-3s} .... r_{0s} r_{D-1d} r_{D-2d})$,

and so on, until it reaches the node with address

$(cd, r_{k-1-Ds} ... r_{0s} r_{D-1d} ... r_{0d})$.

This corresponds to cyclic rotating in the destination's row coordinate one digit at a time. The node reached after $\hat{D}$ hops is the destination node (cd, rd) if and only if

$r_{k-1-Ds} = r_{k-1d}, r_{k-2-Ds} = r_{k-2d}$, and $r_{0s} = r_{Dd}$.

Otherwise, the packet is not in the destination's row rd when it reaches column cd, and the flit takes k more hops to reach its destination (cd,rd). So, a flit reaches its destination with minimum number of hops, which may be either D or $D+k$.

The path of the message through the shufflenet is calculated at the source node and known to the intermediate node when the flit header is received. Instead of storing this information in routing tables which map a particular source and destination node to one of the output channels, we store the complete path information in local virtual channel status table at each node when the flit header is received. The path information stored in the tables consists of a sequence of channels that the message will traverse, along with the number of hops apart from the header flit, i.e. the number of intermediate nodes in the path.

When a message is generated at a node, the path information corresponding to the desired destination is generated according to the above routing algorithm and recorded in the header flit of the message. As the header flit moves through the network allocating the channels for the path, each node simply decrements the hop count and reads the appropriate output channel from the *next* field in the header. Then, the connection is established between the input channel on which the flit was received and the output channel. After reading the output channel, the node removes that information field from the header flit so that, at each node, the appropriate path information is always located in the field of the header immediately following the hop count. This is shown in Fig. 1 for a header flit traveling from source node 1 to destination node 5, requiring three hops. When a node decrements the hop counter to zero, the header flit has reached its destination and the message is to be received by that node continuously. Also, the end of message flag is appended to the last flit of the message. When this is detected by the nodes

in the path, the connection established for the message can be closed and the resources can be allocated to another message.

Since the communication in wormhole routing could be seen as the connection of queues, the sequence of queues traversed by a worm could be used to represent the routing function R for shufflenets.

We present some of the definitions in the following to clearly explain the deadlock-free property of the fixed routing algorithm.

Definition 1: An interconnection network I is a strongly connected directed graph, $I = G(N, C)$. The vertices of the graph N represent the set of nodes. The edges of the graph C represent the set of communication channels. Associated with each channel, ci, is a queue.

Definition 2: A routing function $R : NxN \rightarrow \{C_0, C_1, C_2, ...C_{2k-1}\}$ maps the communication of source node and destination node to a channel sequence traversed by the worm. This sequence length is limited to $2k$.

Definition 3: A channel dependency graph D for a given interconnection network I and routing function R, is a directed graph, $D = G(C, E)$. The vertices of D are the channels of I. The edges of D are the pairs of channels connected by R:

E={$(C_i,C_j)|$ $R(m, n)$={C_0,C_1,....C_{2k-1}} for m,n∈ N, j=i+1, i,j∈ [0,2k-1], C_i, C_j are not empty.}

The authors in [9] then prove a theorem which is the basis for the design of deadlock-free routing algorithms in wormhole routed interconnection networks.

Finally, the reason for mapping each physical channel into 3 virtual channels is that the diameter of the unidirectional shufflenet is $2k$-1. This means that wrap around to the front column is at most twice for any message. Since we have 3 virtual channels in every physical channel, it is enough to route all messages correctly to their destination nodes.

3. Synchronous virtual channel flow control

In bi-directional networks, the asynchronous flow control signals are sent backward simultaneously in all backward channels. Without enough channels, the asynchronous characteristics is changed. In unidirectional networks, the flow control signal is sent by TDM in the common control channel to indicate the state of buffering on the node instead of reverse directional channels simultaneously. All nodes synchronously transmit control information in the control channel and synchronously transmit data according to the received control information and status table information. The architecture of each node and the operation of the flow control are introduced in the following subsections. For convenience, the (p,k) shufflenet is taken as an example for explanation.

3.1. Node structure and modified flit format

In Fig. 2, each node is equipped with one control transmitter, one control receiver, p=2 data transmitters and p=2 data receivers. Each data receiver is associated with one flit buffer. The size of each buffer is only one. Each queue of the receiver is identified by a unique number which is defined according to the input virtual channel. For example, in Fig. 1 each node has six buffers which are noted as {c01, c02, c03; c11, c12, c13}. The buffer's id of node 0 is represented as {0 : **8**, 24, 40; **12**, 28, 44}. The first label is node number. The next six number is separated into two parts because the node degree is two for the (2,k) shufflenet. Each part contains three buffers because each physical channel is shared by three virtual channels. The source buffer (-1) is used to hold the source-generated messages. The size of the source buffer is also one, and other flits are kept by the node memory instead of the node controller. The size of the node memory is assumed to be large enough to hold the source-generated message flits. The buffer's id information is fixed and known to each node in the shufflenet once (p,k) is given. In order to record

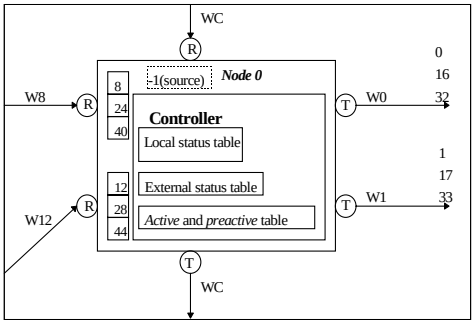

Figure 2. Node architecture.

the information brought by the header flit of the message, each queue is associated with a local status table. The local status table records the information about *path*, *next*, *lookahead*, *hop*, and *state*. The *path* field is decided by the proposed routing mechanism. The *next* field indicates the location of the output channel in the *path* field. The *lookahead* field points to the location of the header flit. It is initialized by the header flit and increased by one once a flit could advance from this virtual channel. The *hop* field is a fixed parame-

ter once the routing is decided. The *state* field records the state of the virtual channel. Only two states can be seen. One is blocking, and the other is free. In addition to these local status tables, the control information from other nodes is recorded in the external status table. The external status table in each node contains all the virtual channel's state information except its local virtual channels. One bit is enough to tell whether the channel is blocking. In order to prevent the collision of the output channel, an *active* table is associated with each data transmitter. This table can indicate whether a channel is used currently. In order to prevent the header flit of one message from conflicting with the end flit of the other message, the *preactive table* is used to record the output channel used by the just-left message. This table records the channel number of just-left messages. Since all flits belong to the same message look to the same location of their header flit, the decision of the state of this message is coherent.

In order to clearly understand the operation of the mechanism of our proposed flow control mechanism, the format of message is introduced below.

All messages are composed of two or three kinds of flits. We classify the flits as header flit, body flit, end flit. Some messages can be very short and contain no body flit. Such messages consist of header flit and end flit only. The content of the header flit must contain the flag and routing information including the hops and the routing path. The flag is used to separate three kinds of flits. The data flit also contains the flag and the output virtual channel. The output virtual channel is derived from the routing path recorded in the local message table. The only difference between the data and the end flits is the flag.

3.2. Node operation

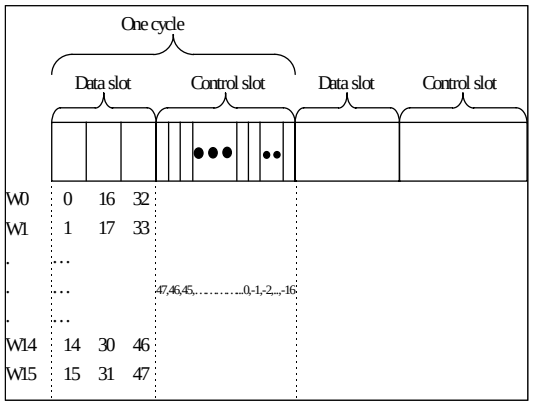

Figure 3. Timing diagram.

In the traditional design, backpressure flow control is asynchronous. The problem of gaps is not serious. The flit can be acknowledged by the backward control signal, and will not continuously transmit data. While in unidirectional scenario, synchronous flow control is necessary. In such flow control, the allocation of channels must be successful in all virtual channels along the message path. Therefore, we must lengthen the control time slot to max number hops of messages, in order to ack all related virtual channels. Taking advantage of the properties of deadlock-free virtual channel routing, where routing algorithms restrict the access order of virtual channels, we let the channel with the largest number ack first, then the second largest , and so on. By doing this, we guarantee that after one control slot all nodes get the correct information.

The operation of synchronous flow control is explained by a simple timing diagram shown in Fig.3. The time is cycle-slotted and each cycle contains a data slot and a control slot. Each data slot is used for transmitted source-generated messages or by-passed messages. All messages in all virtual channels can be transmitted simultaneously but each virtual channel associated with the same physical channel transmits data sequentially in one data slot time. This implementation could simplify the operation of the controller and reduce the buffer size to one. The lower utilization of the data slot is overcome by one modified source routing mechanism. The modified source routing is to change the routing path once there are more choices of the routing path. All the virtual channels are separated into three levels. Level 1 represents channel 0 to channel $p^{k+1}-1$. Level 2 represents channel p^{k+1} to $2 \cdot p^{k+1}-1$. Level 3 represents channel $2 \cdot p^{k+1}$ to $3 \cdot p^{k+1}-1$. Because fixed routing always generate only one channel sequence from level 1, the higher numbered channel in level 2 or 3 is used less frequently. If the routing path generated by the source node is from the low-utilized level 2 or 3, the overall channel utilization is increased. To increase this utilization, the modified deadlock-free source routing is introduced. For example, from node 5 to node 2 the original routing path is -5$\rightarrow$ 10, the modified routing path could be -5$\rightarrow$ 26 or -5$\rightarrow$ 42. This change is done when the controller finds that its original routing path is occupied by another message and the modified routing path is not used by another message. As to the control slot, they need to broadcast their local channel status in our defined order from the higher numbered channel to the lower numbered channel. This order is determined according to the deadlock-free routing algorithm which restricts accessing the channels in the routing path in increasing order. The decreasing order in the control slot makes

the correct status information known to each node in one control slot time. The time is fixed rather than dependent on the length of the routing path. Taking the routing path of some message, -2→ 2→ 12, as an example, the blocking status of channel 12 must be known in advance to channel 2, otherwise the wrong decision may be made by channel 2's status.

3.3. Algorithms and cost analysis

After describing the node architecture and operation, the general algorithm for multihop regular topologies is listed in the following:

Wormhole Routing Flow Control;

```
begin
while (time is not up)
  begin
     control slot(Transmitters & Receivers);
     data slot(Transmitters & Receivers);
  end
end
```

Control Slot(Transmitters);

```
begin
  for the highest channel to the lowest
  channel do
    { corresponding node sequentially
      transmits the status bits of local
      virtual channels.}
end
```

Control Slot(Receivers);

```
begin
  all nodes continue to retrieve the status
  bits from the common control channel and
  store them in the corresponding local
  control tables.
end
```

Data Slot(Transmitters);

```
begin
  for each physical channel do
     checking the active, preactive table,
     local status table and external status
     table in the controller.
     if (transmitted successfully), set the
     active table in the corresponding
     transmission table
     else if (the end flit is advanced), the
     preactive table in the corresponding
     transmission table is used to record the
     output channel of the just-left message.
     else do nothing.
end
```

Data Slot(Receivers);

```
begin
  all nodes receive data flits from their
  corresponding physical channels, decode
  the flits, and record the associated
  information in the local status table.
end
```

The Control Slot(Transmitters) requires that each node know when to broadcast the status of virtual channels. If the topology is regular, this information is easy to obtain. Control slot is composed of $(p \cdot d+1) \cdot N$ subslots. Among them, p is the output degree of each node, d is the number of virtual channels associated with the same physical channel and N is the node number in the topology. For a constant node of shufflenets, d is fixed to 3. With smaller p, the cost of control slot is smaller. Although the smaller p make the diameter of the shufflenet larger, the delay characteristics shown in the simulation result is not changed much in different diameters. The additional hardware requirement is local status table, external status table, and local transmission table. The size of local status table is only proportional to $p \cdot d$, which is a fixed parameter. The size of external status table is proportional to N, but only one state bit is recorded in it. As to the transmission table, the size is $(1 + \log(d \cdot kp^{k+1}))$, which is also a fixed parameter. In the shufflenet example, the hardware cost is kept reasonable.

4. Simulation Model

4.1. Traffic model

We assume that there will be only one message arriving at a node in one cycle time. Since the original Poisson process only considers the arrival rate of messages, the inter-arrival time may be shorter than the message length. Thus, we model the message arrival process as a modified Poisson process whih the leave-to-arrival time being exponentially distributed. The messages arriving at a node are uniformly destined to all the N nodes (excluding the sending station). The length of the message is assumed a variable parameter which is modeled as a geometrically distribution. With the mean arrival rate of the message at a node per cycle and the mean length of the message, the average delay for a message is analyzed. In addition to the delay, the utilization of data slot by the original mechanism is compared with that by the modified source routing mechanism.

Since in our model the queue size is only one, the queue length is not concerned. The *Little*'s formula ($L = W \cdot \lambda$) can not be applied. Furthermore, the modified Poisson process is hard to derive an analytical model and therefore, the simulation is used to simplify the analysis rather than mathematical induction.

4.2. Simulation results

The simulation topology is the (2,k) shufflenet. The simulation is convergent when the simulation time is approaching 10000 cycles. Therefore, our results shown in the following were simulation results for 10000 cycles. The message arrival rate is noted as $Pavg$ which could be viewed as a load indicator. The delay is calculated in average latency cycles from the source to the destination. That is, the delay of the unfinished messages is not calculated.

In Fig. 4, the routing delay increases with the increased message arrival rate and the mean message length. The increased message arrival rate could increase the possibility of blocking and therefore the delay is increased. Since the delay is calculated from source to destination, longer message length will cause longer delay. But longer delay does not necessarily generate poor throughput since it is dependent on the utilization of data slot. For the same message length with lower message arrival rate, the blocking probability is low and thus the delay is increased little. But with higher message arrival rate, the delay is increased much. This gives us the information that when the load ($Pavg$) is high, the mean message length could be made shorter.

In order to observe the delay characteristics of shufflenets in topologies with different diameters, we compare the (2,3), (2,5) , (2,10) shufflenet for the mean message length equal to 100 in Fig. 5.

The delay in variable diameters of the shufflenet topology is almost the same. This result tells us that the delay is not dependent on the routing path and therefore, the benefit of wormhole routing is obtained. Since the delay in the traditional store-and-forward routing increases with the diameter of the topology, the stable delay characteristics of our wormhole routing would be better.

The bi-directional network could have shorter delay and larger throughput. The shorter delay comes from the bi-directional communication and the larger throughput comes from the more available channels. With limited resources, such as the number of wavelengths and transceivers, the proposed unidirectional flow control could obtain reasonable delay and throughput while the bi-directional network can lose its function due to no backward channels. In order to improve the throughput, the modified source routing is used to increase the channel utilization. Higher channel utilization means higher throughput. In Fig. 6, the original and the modified routing mechanisms are compared with mean message length equal to 100.

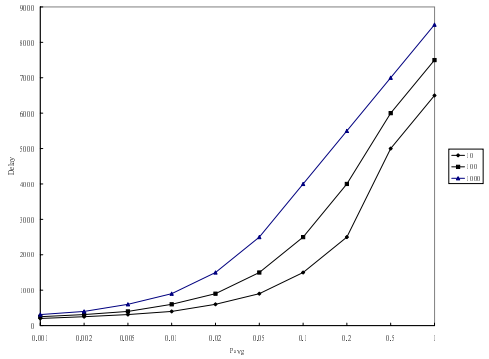

Figure 4. Delay results for the (2,3) shufflenet.

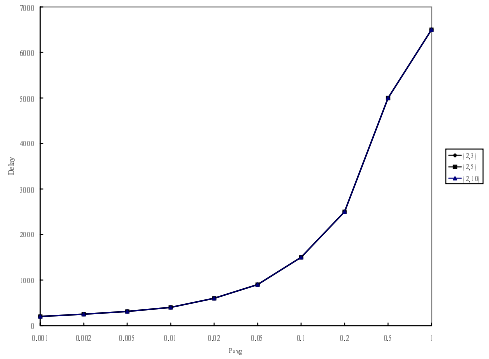

Figure 5. Delay for different diameters of the shufflenet topology.

After modifying the source routing, the utilization is increased no matter what the $Pavg$ is. Further, from the simulation results we see that the utilization is saturated for $Pavg \geq 0.1$. This is because the blocking effect dominates the simulation. This utilization could not reach 100

5. Conclusions

In this paper, we have proposed a new wormhole routing flow control mechanism for incorporating the benefit of wormhole routing into an unidirectional optical network. The synchronous flow control is oper-

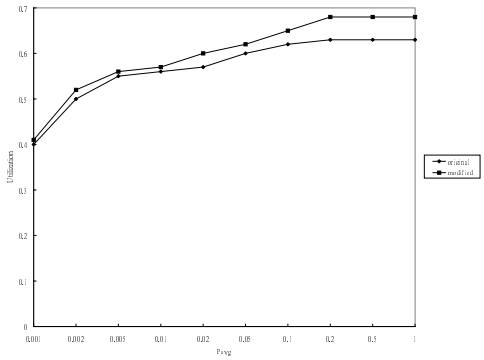

Figure 6. Utilization comparison between original and modified source routing.

ated in the common control channel instead of the distributed multiple backward channels. The size of the synchronous control slot is not dependent on the path length and is only linearly proportional to the node number. For example, the size of control slot is only 7*node number bits in a $(2,k)$ shufflnet.

The simulation results indicate that the routing delay is stable no matter what the mean message length or the mean diameter of a shufflenet is. This shows that the benefit of shorter routing delay is obtained since in the original store-and-forward routing the routing delay increases with the size of mean diameter. The smaller buffer size requirement of wormhole routing is significantly reduced to one flit size according to the simultaneous serving protocol. The low utilization of data slot induced by the channel serving protocol is overcome by the modified source routing. The simulation result shows that the utilization of data slot in the modified source routing is increased compared to the original source routing.

References

[1] L. Kleninrock et al., "OPTIMIC: A Scalable Distributed All-optical Terabit Network," *J. High Speed Networks*, vol. 4, no. 4, 1995, pp. 407–424.

[2] L. Kleninrock et al., "The Supercomputer Supernet Testbed: A WDM Based Supercomputer Interconnect," *Joint issue, IEEE JSAC and IEEE/OSA J. Lightwave Tech.*, vol. 14, no. 6, June 1996, pp. 1388–99.

[3] N. Boden et al., "Myrinet: A Gigabit-per-Second Local-Area Network," *IEEE Micro*, vol. 15, no. 1, Feb. 1995.

[4] B. Mukherjee, "WDM-based Local Lightwave Networks Part I: Single-Hop Systems," *IEEE Network*, pp.12–27, May 1992.

[5] B. Mukherjee, "WDM-based Local Lightwave Networks Part II: Multi-Hop Systems," *IEEE Networks*, pp12–27, May 1992.

[6] M. Gerla, et al., "Protocols for an Optical Star Interconnect for High Speed Mesh Networks", *Proceedings of GLOBECOM 95*, pp. 146–152, June 1995.

[7] P. Palnati, M. Gerla, and E. Leonardi, "Deadlock-Free Routing in an Optical Interconnect for High-Speed Wormhole Routing Networks," *Proc. 1996 Int'l. Conf. Parallel and Distributed Sys. (ICPADS)*, pp. 256–264, June 1996.

[8] E. Leonardi, et al. "Congestion Control in Asynchronous, High-Speed Wormhole Routing Networks," *IEEE Comm. Mag.*, pp. 58–69, Nov. 1996.

[9] W. Dally and C. Seitz, "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks," *IEEE Trans. on Computers*, C-36(5), pp. 547–553, May 1987.

[10] M. J. Karol and S. Shaikh, "A Simple Adaptive Routing Scheme for ShuffleNet Multihop Lightwave Networks," *INFOCOM 88*, pp.1640–1647.

[11] B. Li and A. Ganz, "Virtual Topologies for WDM Star LANs - The Regular Structures Approach," *INFOCOM 92*, pp. 2134–2143.

[12] M. G. Hluchyj, M. J. Karol, "ShuffleNet: An Application of Generalized Perfect Shuffles to Multihop Lightwave Networks," *IEEE J. Lightwave Tech.*, vol. 9, no. 10, pp. 1386–1397, Oct. 1991.

[13] W. J. Dally, "Virtual-Channel Flow Control," *IEEE Trans. on Parallel and Distributed Systems*, vol. 3, no. 2, pp. 194–205, Mar. 1992.

[14] N. Maxemchuk, "Routing in the Manhattan Street Network," *IEEE Trans. Communications*, vol. 35, no. 5, pp. 503–12, May 1987.

[15] T. Rodeheffer, "Experience with Autonet," *Computer Networks and ISDN Systems*, 25(6):623–629, Jan. 1993.

[16] L. M. Ni and P. K. McKinley, "A Survey of Wormhole Routing Techniques in Direct Networks," *IEEE Computer*, pp. 62–76, Feb. 1993.