Reducing Communication Latency with Path Multiplexing in Optically Interconnected Multiprocessor Systems

Chunming Qiao, *Member*, *IEEE Computer Society*,
and Rami Melhem, *Member*, *IEEE Computer Society*

Abstract—Reducing communication latency, which is a performance bottleneck in optically interconnected multiprocessor systems, is of prominent importance. A conventional approach for establishing connections in multiplexed networks uses a set of *independent* time slots (or virtual channels) along a path for each connection. This approach requires the use of switching devices capable of interchanging time slots, and thus introduces latency in addition to hardware and control complexity. In this paper, we propose an approach to all-optical *Time Division Multiplexed* (TDM) communications in multiprocessor systems. The idea is to establish a connection along a path using a set of time slots (or virtual channels) that are *dependent* on each other, so that no time-slot interchanging is required. We compare the proposed approach with the conventional one in terms of the overall communication latency. We found that, despite the possibility that establishing a connection may take a longer time, the proposed approach will result in lower *overall* communication latency as it eliminates the delays introduced by the time-slot interchanging switching devices.

Index Terms—Communication latency, fiber-optical interconnects, switching networks, time division multiplexing, time slot interchangers.

1 Introduction

Optical interconnects offer many advantages over its electronic counterpart including high connection density and relaxed bandwidth-distance product [1], [2], [3]. They are now widely accepted by telecommunications industries and have also been considered in the design of commercial multiprocessor systems to reduce the wiring density and to increase the system scalability [4], [5]. We believe that *Time-Division Multiplexed* (TDM) communications in optically interconnected multiprocessor systems is a way to achieve significant advancement in performance beyond the state-of-the-art.

Time-Division Multiplexed (TDM) communication techniques are useful in creating multiple *virtual channels*, each of which corresponds to a *time slot* on a single *physical* link in interconnection networks. Having virtual channels increases bandwidth utilization and facilitates *adaptive routing algorithms* as well as the static mapping of the communication requirements of various applications [6], [7], [8]. *TDM* techniques are also useful for tolerating the propagation latency in optically interconnected multiprocessor systems [9], [10], [11], [12], and for reducing the control complexity of channel allocation in TDM systems [13], [14], and in *Wavelength Division Multiplexed* (WDM) systems [15], [16].

A key issue to be addressed in optically interconnected multiprocessor systems is the reduction of the *communication latency* which is a performance bottleneck in such systems. In this paper, we describe a new multiplexing approach for establishing all-optical connections in multiprocessor systems, and compare it with a conventional multiplexing approach.

The paper is organized as follows. In Section 2, we provide motivations for considering circuit-switching, and describe possible ways to achieve global synchronization which is required for TDM communication. In Section 3, we describe how connections, especially virtual connections, are established in multiplexed networks. Specifically, a conventional approach, which we call *Link Multiplexing* (or LM), is described in Section 3.1. Using LM, a connection may be established by selecting a time slot on each link *independently* of the time slots selected on the other links along a path. Thus, in order to transfer messages between two possibly different time slots, the switches in the network are required to have the capability of interchanging time slots [17], [18], [19]. The proposed approach, called *Path Multiplexing* (or PM), is described in Section 3.2. Using this approach, the time slots selected on the links along a path are *dependent* on each other such that no time-slot interchanging is needed to transfer messages along the path. For example, a connection may be established along a path by selecting the same time slot on every link. Network control, or signaling, involved in selecting the time slots to establish a connection is described in Section 3.3.

In Section 4.1, we examine the components of the overall communication latency and compare these components in the two approaches. Intuitively, given a network containing a set of existing connections, it is more likely that a set of

- *C. Qiao is with the Department of Electrical and Computer Engineering, State University of New York at Buffalo, Buffalo, NY 14260. E-mail: qiao@photon.eng.buffalo.edu.*
- *R. Melhem is with the Department of Computer Science, University of Pittsburgh, Pittsburgh, PA 15260. E-mail: melhem@cs.pitt.edu.*

Manuscript received May 26, 1995.
For information on obtaining reprints of this article, please send e-mail to: transpds@computer.org, and reference IEEECS Log Number D95238.

independent time slots, rather than a set of time slots that are *dependent* on each other, is available for establishing a new connection. Therefore, the conventional *LM* may result in shorter *blocking time*. However, time-slot interchanging in LM introduces delays as a part of network propagation latency. Thus, PM may reduce the *overall* communication latency, as well as the hardware and control complexities due to the elimination of time-slot interchanging. In the rest of Section 4, we present both analytic and simulation results which show that, for the range of parameter values we considered, the proposed PM approach reduces the overall communication latency when compared to the LM approach. Thus, PM may be used in the design of high performance, low complexity optically interconnected multiprocessor systems. We conclude this paper in Section 5.

2 Communications in TDM Networks

We consider a network which consists of switches having a fixed number of inputs and outputs. Some of these inputs and outputs are used to interconnect with other switches through a set of *external* links, while the others are used to connect to a local processing element through a set of internal links. Fig. 1 shows an example with a mesh-like topology. Each switch in this network has four pairs of external links to adjacent switches and one pair of internal links to a processing element (PE). Each of these links can be time-multiplexed to create multiple virtual links in multiple time slots. As high bandwidth is easier to obtain with optical interconnects than with electronic ones [1], [2], [3], we propose to use fiber links (connected with optical switches) as an alternative to support multiprocessor communications.

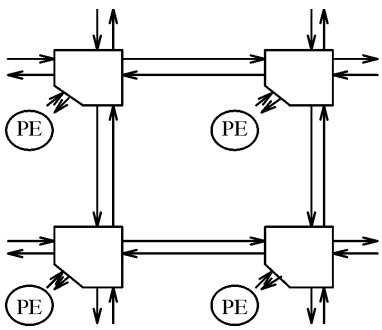


Fig. 1. A direct network with a mesh-like topology.

2.1 Circuit Switching

Processors communicate with each other by sending and receiving messages, which can be done via either *circuit-switching* or *packet-switching*. In this paper, we study circuit-switched networks, and, in particular, focus on the communication latency in the two TDM circuit-switching methods, LM and PM, which will be described in details in Section 3.1 and Section. 3.2, respectively. What motivates this study is that with optical interconnects, packet-switching would require conversions between electronic and optical signals at intermediate switching nodes for address decoding purposes. Such conversions are expensive in both time and implementation cost. On the other hand, since an optically interconnected system has enough bandwidth, *circuit-switching* can be used to trade in some bandwidth for the elimination of such conversions. In addition, with high available bandwidth, software (or protocol) latency and/or hardware (or network communication) latency become the performance-limiting factors. Using light weight protocols such as compiled communications [20], [21], and memory mapped interfaces [22], [23], [24], software latency can be significantly reduced. In such cases, it is important to reduce the network communication latency.

2.2 Synchronization

Optical clock distribution systems offer less clock skew and better noise immunity than electronic ones. The relatively larger fan-out of optics makes it possible to distribute a global clock generated by a high-power laser to hundreds of receiver modules at a high frequency [25]. An optical clock distribution system in which each processor or switch receives an identical copy of the global clock signal was recently implemented in the Cray T90 Series with up to 32 processors.

We consider a TDM network, in which global synchronization is required at the data packet level, not at the bit level. More specifically, multiplexing of different messages is done based on time slots, and the duration of a time slot is typically equal to the duration of several hundreds of bits. For synchronization purposes, a guard band at each end of a time slot can be used to accommodate possible drifting or jitters. For example, if the duration of a time slot is 276 *ns*, which includes a guard band of 10 *ns* at each end, then 256 *ns* can be used to transmit data. If the transmission rate is 1 *Gb/s*, then a packet having 256 bits can be transmitted during each time slot. The issues related to the optimal duration of a time slot, which depends on many factors including the transmission technology as well as the communication requirements of the applications, will not be addressed in this paper.

In a TDM network, each physical link is time-multiplexed at the rate of *time slots*. Specifically, the time horizon on each link is divided into intervals of K time slots, for some K. Each time slot corresponds to a virtual channel, and each interval of K time slots is referred to as a *frame*, where K is called the frame size, or the *multiplexing degree*. Typically, the time for a signal to travel each link (at most a few feet) is less than the duration of a guard band. Hence, the processors and switches in the network can be synchronized under a global time slot clock. A connection can be established by using a set of time slots, one on each link on a physical path from a source to a destination. For a *given link*, if time slot i ($0 \le i \le K-1$) of a frame is used to establish a connection, then for the duration of the connection, the same time slot of the *next frame* is also used for that connection. However, as will be discussed later, different time slots may or may not be used on *different links* depending on whether LM or PM is used.

3 Establishing Virtual Paths

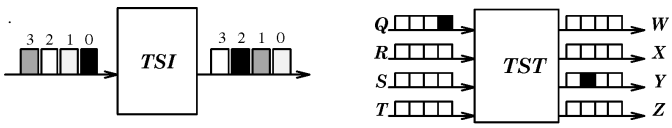
In order to establish a connection, a physical path (*PP*) from a source to a destination is chosen first. Afterwards, a virtual path (*VP*) consisting of several time slots, one on each

link of the physical path, is selected and the connection is established. How a *PP* can be selected has been studied extensively and is well understood [26]. In what follows, we illustrate how a connection can be established assuming that a *PP* has been selected. Specifically, in the following two subsections, 3.1 and 3.2, we give detailed descriptions and examples of how time slots should be selected on two adjacent links, and what kind of switches are required using the conventional *Link Multiplexing* (LM) and the proposed *Path Multiplexing* (PM) approaches, respectively. Then in Subsection 3.3, we describe network control or signaling, that is, how control information needed to establish a *VP* is exchanged.

3.1 Link Multiplexing (LM)

One way to establish a *VP* in a TDM network is the so-called *Link Multiplexing* (LM) approach. More specifically, a *VP* may be established over a set of *independent* time slots along a *PP*. This, however, calls for the use of *Time Slot Interchangers* (TSIs) in order to transfer the information carried over the same connection from one time slot on one link to another time slot on the next link along the path.

A TSI capable of interchanging K (> 1) time slots can be viewed as a "black box" having one input and one output, each being multiplexed with K virtual channels, or equivalently, each carrying frames of K time slots. In Fig. 2a, K is assumed to be 4, and thus an input or output frame has four time slots. The TSI switches time slots 0, 1, 2, and 3 of the input frame to time slots 2, 0, 3, 1 of the output frame, respectively. By setting the TSI properly, any time slot of the input frame can be switched to any *idle* time slot of the output frame (i.e., any time slot to which no other time slots of the input frame has been switched).



(a). A Time Slot Interchanger (TSI). (b). A Time-Space-Time (TST) switch.

Fig. 2. TSI and TST switches.

A more generalized form of a TSI is the so-called *Time-Space-Time* (TST) switch which has $S \geq 1$ inputs and outputs. When $S = 1$, a TST switch becomes a TSI. When $S > 1$, a TST switch may be viewed as an *augmented* $S \times S$ switch with each input and output being multiplexed with K time slots. By setting a TST properly, any time slot of any input can be switched to any idle time slot of any output. Fig. 2b shows a TST with $S = K = 4$, in which time slot 0 of input Q is switched to time slot 2 of output Y.

We now illustrate the *LM* approach using an example. As shown in Fig. 3, each switch in a network is required to be a TST switch. Assume that a *VP* needs to be established on two physical links X and Y. Since switch α can interchange time slots, the message may arrive during any time slot i on X and then leave during any time slot j on Y, where $0 \leq i, j \leq K - 1$, as long as these two time slots are not being used by other connections. That is, when establishing the connection using LM, time slots i and j will be selected *independently* on links X and Y, respectively.

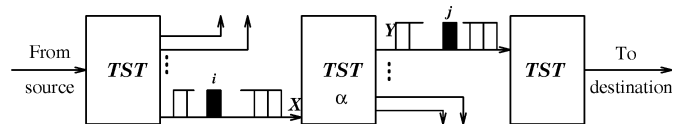


Fig. 3. Establishing a connection with LM.

3.1.1 Electronic TSIs

A TSI (or TST) can be implemented either electronically or optically. An electronic implementation of a TSI for $K = 4$ is shown in Fig. 4. An Optical-to-Electronic (O/E) conversion circuit (i.e., receiver) at the input side converts the incoming optical signals to electronic ones. The converted signals from time slots 0, 1, 2, and 3 are then stored in input buffers 0, 1, 2, and 3, respectively. The 4×4 crossbar switch between the input and the output buffers is then set such that the content of a given input buffer is copied to a desired output buffer (Fig. 4 shows the setting used in the TSI in Fig. 2a). Finally, an Electronic-to-Optical (E/O) conversion circuit (transmitter) transmits the content of the output buffers 0, 1, 2, and 3 in that order.

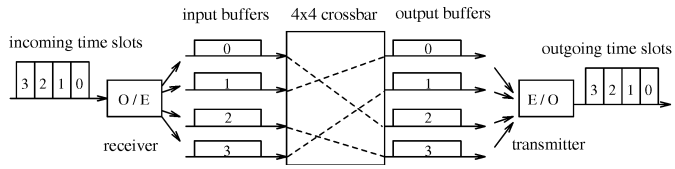


Fig. 4. An electronic implementation of a TSI.

One of the drawbacks of an electronic implementation of a TSI (or TST) is the expensive high speed receiving and transmitting circuits inside each switch, which also makes the TSI (or TST) nontransparent to bit-rate. Specifically, in a network consisting of $S \times S$ TSTs, in addition to one pair of receiving and transmitting circuits at the processor-to-network (or processor-to-switch) interface, another $S - 1$ pairs are needed inside each switch, one for each input-output link. Since these circuits are designed for a specific receiving and transmitting speed (say at 1 Gb/s), one will have to replace all of them to achieve *higher* bit-rate transmissions (say at 2 Gb/s).

Another drawback that is of special interest to us in this paper is the delay introduced in the process of buffering (and interchanging) time slots because both the input and output frames need to be aligned according to a frame clock. Specifically, if *frame integrity* is desired, that is, all the incoming time slots belonging to an input frame need to be in the same output frame, then every incoming time slot will need to be delayed by at least K time slots. Some of the incoming time slots will be delayed additionally because of the need to change their relative positions in the output frame. For example, incoming time slot 0 will be delayed for 4 + 2 = 6 time slots in order to become outgoing time slot 2. Even if frame integrity is not desired, an incoming time slot may also have to be delayed. For example, in Fig. 4, since incoming time slot 1 needs to become outgoing time slot 0, it has to be transmitted in the next frame, resulting in a delay of three time slots.

3.1.2 Optical TSIs and TSTs

A TSI (or TST) can also be implemented *optically* using fiber-loop delay lines and Lithium Niobate directional couplers (also called *switches*) [17], [27], [19]. Fig. 5 shows an optical TSI for $K = 4$. The TSI consists of five 2×2 switches, each of which can be set to either straight ("=") or cross ("X") (the symbol "?" in the figure stands for "don't-care"). A fiber-loop delay line with one time slot delay is placed on the lower link between the first and second switches as well as between the fourth and fifth switches. In addition, a delay of two time slots is placed between the second and third switches as well as between the third and fourth switches. With each of the five switches set to its sequence of states as shown in the figure, incoming time slots 0, 1, 2, and 3 become outgoing time slots 2, 0, 3, and 1 after being delayed for 4, 1, 3, and 0 time slots, respectively (Fig. 5 shows the setting used in the TSI in Fig. 2). Note that in this implementation, the output frame is not aligned with the input frame. However, because incoming time slot 3 needs to become outgoing time slot 1, the output frame can not start until time 2. In other words, the output frame needs to be delayed by two time slots. If incoming time slots 0 and 3 need to become outgoing time slots 3 and 0, respectively, then incoming time slot 0 should be delayed for a maximum delay of six time slots, which is why the total number of delays available in the TSI is six.

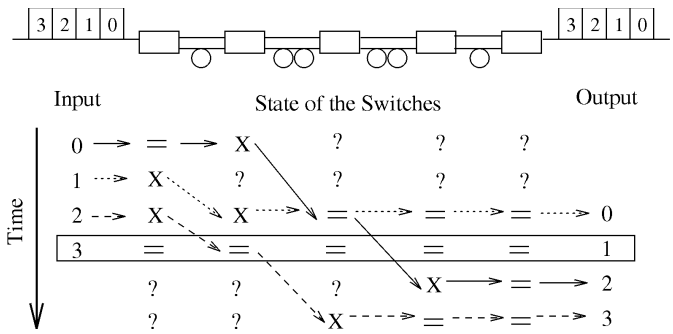


Fig. 5. An optical implementation of a TSI.

Note that although such an optical implementation eliminates E/O and O/E conversions during the switching, therefore providing *bit-rate transparency*, optical TSIs are not without drawbacks. For one thing, they, like electronic TSIs, will introduce delays as a part of propagation latency of a network as illustrated above. In addition, not only are the control and hardware more complex, optical TSTs will introduce additional crosstalks and attenuations of optical signals. These undesirable effects could be detrimental to the performance of optically interconnected multiprocessor systems. In the next section, we describe an approach that can establish *all-optical* connections without using TSTs.

3.2 Path Multiplexing (PM)

Before we describe the proposed approach in details, we first review a connection paradigm, called *Reconfiguration with TDM* (RTDM), which was previously developed for indirect networks [13], [14].

Using the RTDM, an indirect network, such as a crossbar or a Multistage Interconnection Network (MIN), behaves like a *Time-Multiplexed Switch* (TMS). More specifically, the network repeatedly goes through a sequence of configurations, one during each time slot. Fig. 6 illustrates a possible sequence of four (4) configurations that a 4×4 switch may go through. Like the TST switch shown in Fig. 2b, each input and output of the TMS is multiplexed with degree $K = 4$. However, unlike that TST, a time slot i of any input link is switched to the same time slot i of an output link in the TMS, as shown in the figure. This means that such a TMS does not have the capability of interchanging time slots as a TST switch does. As a result, the signals arriving at a given input during time slot i will go to an output which depends solely on the *switch setting* during that time slot.

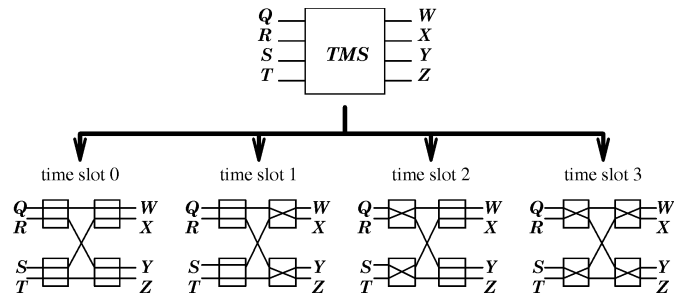


Fig. 6. Connecting inputs to outputs in TMS.

In an $S \times S$ switch, all possible S^2 connections can be established by multiplexing the switch with degree $K = S$ [28], [29]. However, this may result in an unnecessarily long latency because a given input-output connection is established once every S time slots. *RTDM* provides efficient multiprocessor communications through either *static* [13] or *dynamic* [14] reconfiguration which reduces the multiplexing degree K by establishing only those required connections. Note that one can easily adapt K in a TMS according to application requirements [13], [14], whereas, in an optical TST, K is fixed by the architecture. The proposed *Path Multiplexing* approach extends the principle of RTDM to establishing a connection in a direct network consisting of TMS switches multiplexed with degree K.

Fig. 7 illustrates the idea of the PM approach. Let us assume that a *VP* needs to be established along links W, X, and Y, and that switch α can be set to interconnect link W with link X during time slot i ($0 \le i < K$). Consider the case in which the propagation delay on link X is *negligible* when compared to the duration of a time slot. Since any message carried on the connection will arrive at switch β also during time slot i, we have to be able to set switch β so that link X is interconnected with link Y during time slot i. (in the figure, this means $j = i$). Note that if the link propagation delay on X is *not* negligible (say, equal to L_P time slots), then the connection can be established only if during time slot $j = (i + L_P)$ *mod* K, switch β can be set so that link X is interconnected with link Y. In either case, the time slot to be used on link Y (which is j) is dependent on the time slot used on link X (which is i). In general, when using the PM approach, the time slots on the links along a path are selected in such a way that packets can flow through the links without being buffered as no time slot interchanging is required.

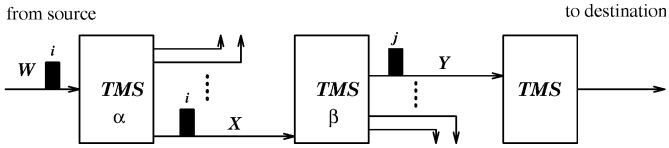


Fig. 7. Establishing a connection with PM.

3.3 Network Control

In this section, we describe how a connection can be established, assuming that a physical path has already been chosen. We assume that in addition to the data network, which uses time-multiplexed optical interconnects, there is a *control network* which is used to exchange control information needed to establish virtual (and physical) paths. Although the control network may have a physical topology different from the data network, we will assume that the same physical topology is used to interconnect *routers* (or communication coprocessors) that are attached to the switches. As will be discussed next, the amount of bandwidth required for exchanging control information is relatively low compared to that required for exchanging data. For example, the address of a node in a system having 1,024 nodes is only 10-bits long. A control packet may thus be about 20 to 30 bits long, which is about one-tenth of a data packet (assumed to contain 256 bits). Thus, the control network may operate at a bit-rate that is much lower than the data network, which makes it feasible to implement the control network in electronics.

Many techniques developed for establishing a *PP* in a direct network can be applied to establish a *VP*. One way to reserve / release a *PP* in a hypercube was described in [30]. A way to reserve / release a connection (including a *PP* and a *VP* according to the RTDM paradigm) in MINs was described in [13]. Although there are many variations as to how a *VP* can be established, we will concentrate only on a basic scheme under distributed control in the following discussions. In particular, we will describe a scheme for PM as the case for LM is similar.

Let the set of available time slots on a link *L* be maintained in a list denoted by *Avail*(*L*). This list can be implemented as a *K*-bit string, where *K* is the number of time slots in a frame. In addition, there are two lookup tables called the *PP*-table and the *VP*-table, which store information about the links that form a *PP* and the time slots that are assigned to a *VP*, respectively. These information are distributed among the routers. Specifically, each router has access to the *Avail*(*L*) lists of all the *outgoing* links of the switch it is attached to. It also has access to the part of the *PP*-table which specifies which of these outgoing links to use for a given *PP*, and the part of the *VP*-table which specifies which time slot on a particular outgoing link has been assigned to a given *VP*.

Control information can be exchanged among the routers in the forms of packets for reserving and releasing *VP*s as follows: A source constructs a reservation packet *Res_VP*, buffers a copy of the request, sends the request to its local router, and waits for an acknowledgment (either positive or negative). The *Res_VP* packet specifies the source and destination addresses or simply the ID of a *PP*. It also contains a list of time slots, denoted by *Init* (also implemented as a *K*-bit string), during which the processor wishes to transmit.

Upon receiving the *Res_VP* packet, a router identifies the outgoing link *L* based on its PP table. The router then performs the following "set-intersection" operation $Init = Init \cap Avail(L)$ (implemented with a bit-wise *AND* operation), which updates the list in the packet. If the resulting *Init* list is not empty, then a part of the *Avail*(*L*) list, which includes the set of time slots in *Init*, will be "locked" to prevent them from being used by other *VP*s. The *Res_VP* packet is then forwarded to the next router along the *PP* until the destination is reached.

When *Res_VP* reaches its destination, a time slot from its *Init* list, denoted by *ts*, is selected by the destination processor, which constructs an acknowledgment packet, *Ack_VP*, containing *ts*. The *Ack_VP* packet is then sent back to the source processor in the direction opposite to the one taken by *Res_VP*. Upon receiving *Ack_VP*, each router "unlocks" the part of *Avail*(*L*) list previously "locked" by *Res_VP*. Time slot *ts* is deleted from that part (and from *Avail*(*L*)) and assigned to the *VP*. The router also sets the switch it is attached to so that data carried on the *VP* will be switched to time slot *ts* on link *L*. Finally, *Ack_VP* is sent to another router until the source is reached.

Upon receiving *Ack_VP*, the source can start message transfer at the beginning of the next transmitting time slot, which may or may not be during the current frame. If a message is divided into *m* data packets, then the message transfer will be completed after *m* frames since one data packet is sent in a time slot during each frame. Upon the completion of the message transfer, the source constructs a release packet, *Rel_VP*, and sends it to the destination. The time slot assigned to the *VP* can then be released.

If the resulting *Init* list in a *Res_VP* packet becomes empty at an intermediate router after the "set-intersection" operation, the router will drop *Res_VP*, and construct an *Nack_VP* packet. The *Nack_VP* packet will be sent to the source in a way similar to that an *Ack_VP* is sent. The difference is that this time, only the "unlock" operation will be performed at each router. After the source receives *Nack_VP*, it may send a copy of the *Res_VP* packet at a later time.

We note that signaling can be done in LM in a similar way, except that there is no need to maintain the *Init* list or perform the "set-intersection" operation. A *Res_VP* packet will "lock" *an arbitrary time slot* from the *Avail*(*L*) list at each intermediate router. These locked time slots will be "unlocked" by either an *Nack_VP* or an *Ack_VP* packet, as in PM. In the latter case, the time slots previously locked are then assigned to the *VP*.

We also note that when the control network operates at 100 Mbs, which is one-tenth of the bit-rate of the data network we have assumed (i.e., 1 Gbs), the time needed to transmit a control packet is a few hundreds of nanoseconds, which is about one time slot. Assuming that processing a control packet at a router also takes a few hundreds of nanoseconds, then the time for a control packet to go from one router to the next is about two time slots. The maximal time a source has to wait before it receives an acknowledgment is about $4H$ time slots, where H is the number of hops

between the source and destination. This amount of time is a modest overhead when compared to the time for a message to be received by the destination.

4 Comparing PM with LM

Since TST switches have higher control and hardware complexity than TMS switches, it is more expensive to adopt the LM approach than the proposed PM approach, especially if the former uses electronic TSTs. In addition, when it comes to choosing an appropriate multiplexing degree for an application, the PM approach is much more flexible than the LM approach especially if the latter uses optical TSTs. Complexity and flexibility aside, what we will focus on in this comparative study, is the performance of these two approaches in terms of the communication latency they will introduce. As mentioned earlier, it may take longer to be able to establish a VP in PM than in LM because a request that has to reserve the same time slot on different links (as in PM) is more likely to be blocked than a request that can reserve possibly different time slots (as in LM). On the other hand, since the TMSs used in PM do not introduce delays for buffering and interchanging time slots, data can flow through the network with less delay in PM than in LM, once a *VP* is established. Hence, the overall communication latency in the PM approach may be smaller than that in the LM approach. In the next three subsections, we first examine the components of the communication latency, and then compare the PM and the LM approaches via analysis and simulations.

4.1 Communication Latency

In circuit-switching, the communication time includes both message-transmission time and *communication latency*. If a message is divided into m packets (of 256 bits each), then the time required for transmitting the message into the network is Km time slots in either LM or PM, as described in Section 3.3. For a large message size (i.e., a large m), the bandwidth, not the latency of the network, is important. On the other hand, for a small message size, the latency becomes more important. It is well known that optical interconnects are suitable for high-bandwidth communications, that is, when the message size is large. One of our focuses in this paper is to study ways to reduce latency in optically interconnected systems where the message size is small (say between 512 bits and 2,048 bits, or $2 \le m \le 8$). Since the message-transmission time is the same in LM and PM, it will not be considered in this comparative study of LM and PM.

The communication latency is the sum of two component values: *circuit-establishment latency*, L_E and *network-propagation latency*, L_N. The former includes *blocking time*, denoted by L_B, and *signaling overhead*. The blocking time is largely due to the intervals between the time a request fails and the time the failed request is resubmitted. Specifically, it is the duration of the period from the time a connection request is first generated by a source till the time the first packet can be sent over the established connection, excluding the signaling overhead, which is the amount of time to exchange control information. Since the signaling overhead depends on many factors such as the network topology and the control scheme used (e.g., either centralized or distributed) is about the same for both LM and PM, it will not be considered in this comparative study of LM and PM. In other words, we will use L_B in place of L_E.

The second component of the communication latency, L_N, is the time for a signal to traverse an established path from a source to a destination. Like the circuit-establishment latency, L_N is further composed of two parts: *link propagation* latency and *switching* latency. The first part is the time for an optical signal to traverse the external links along a path. Since in a reasonable network topology, even the longest path has less than a few tens of hops, and since an optical signal can traverse each hop in much less than a nanosecond in a multiprocessor system, the link propagation latency is often *negligible* relative to the duration of a time slot (a few hundreds of nanoseconds as discussed in Section 2).

The second part of L_N, the switching latency, is the time for a message to go through the switches along a path. If PM is used, then this part is also negligible since an optical signal traverses a switch in the optical domain without being buffered. Accordingly, L_N can be ignored if PM is used. However, as discussed in Section 3.1, a TST switch capable of interchanging time slots introduces a delay because a signal has to be buffered in either the optical or the electronic domains, which could vary between 0 and $2K - 1$ time slots depending on the implementation of the TSTs as well as on how the time slots need to be interchanged. For simplicity, we assume that each incoming time slot will be delayed, on average, by K time slots when going through a TST. Assuming that the distance between a source and a destination is H hops, the total switching latency along the path, excluding the source or destination points, is $(H - 1) \times K$. Accordingly, if LM is used, L_N, which is about $(H - 1) \times K$, can no longer be ignored.

To summarize, if we let the average *blocking time* in PM and LM be $L_B(PM)$ and $L_B(LM)$, respectively, and we denote the *communication latency* in the two approaches by $L(PM)$ and $L(LM)$, respectively, we have

$$L(PM) = L_B(PM) \tag{1}$$

$$L(LM) = L_B(LM) + (H - 1) \times K \tag{2}$$

Throughout our comparative study of the LM and PM approaches, we will focus on the value of $L(PM)$ as defined in (1) and the improvement ratio, I, which is defined as follows:

$$I = \frac{L(LM) - L(PM)}{L(LM)} \times 100\% \tag{3}$$

Hence, I will be at most 100% when $L(PM) = 0$. The larger I is, the better PM becomes when compared to LM. For example, $I = 50\%$ means that $L(PM) = L(LM)/2$ while $I = 75\%$ means that $L(PM) = L(LM)/4$.

Our comparative study involves two parts: analytical work and simulation. In both cases, a direct network whose links are time-multiplexed with degree K is studied and the communication latency resulted from LM and PM is evaluated. Common assumptions to both the analysis and simulation that

simplify our evaluation processes are made. These assumptions are:

1) A predetermined *shortest path* is used to establish a given connection (even though alternate paths may exist).
2) Each PE has a request buffer of size b. During each time slot, if the buffer is not full, then the PE generates, with probability r, a new request to transmit a message of average length of m packets, $m \geq 1$. Thus, the message (request) generation rate is r, and the traffic (packet) rate is $r' = m \cdot r$.
3) A copy of the rejected request is resubmitted after t time slots, where t is a multiple of the frame duration, K. Note that our previous study in [14] has shown that *normalized service time* [31] is minimized when K is around 3 to 5. Hence, we will focus on the results obtained with a small K.
4) The destination of each message is randomly chosen among all the processors.

We will elaborate on these assumption as we go along.

4.2 Probability Analysis

In this section, we study the probability that a requested connection can be established using either PM or LM when the network is in a steady-state. This probability will then be used to calculate L_B. We consider an $N \times N$ torus in which each switch has four pairs of external links, and the average length of a connection in terms of number of hops is about $N/2$.

We will first consider the case in which there is no time-multiplexing. In order to facilitate our analysis, we assume that the traffic is evenly distributed in the network. Hence, the traffic at all switches is statistically identical. Moreover, at each switch, the traffic on the input (or output) links is also statistically identical.

We will use the model illustrated in Fig. 8 to study the steady-state behavior of the network. We denote by u the probability that a given time slot on any external link is occupied by a connection. In addition, we denote by v the probability that a given time slot on the internal link from a PE to its switch is occupied by a connection (originating from PE). Then, the probability that a given time slot on the other internal link (from the switch to the PE) is occupied by a connection (destined to PE) is also v.

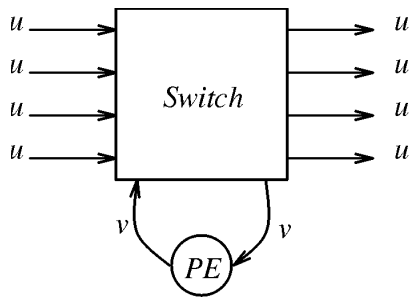


Fig. 8. A switch in a steady-state torus network.

Let H be the average length of a connection (in terms of number of hops). Since the total number of established connections is vN^2, each occupying an average of H links, and there are $4N^2$ links in a torus, the probability that a given link is occupied is $u = vN^2 H / 4N^2$. That is,

$$u = vH/4 \tag{4}$$

Alternatively, the above relationship between u and v can be derived by treating u and v as some fraction of a connection carried on an external input link and an internal output link, respectively. Since the probability that a given connection will terminate at any switch is $1/H$, we have $v = 4u/H$.

We will use the notation $P[H, K]$ (the success probability) or $\overline{P}[H, K]$ (the failure probability which equals $1 - P[H, K]$) to denote the probability that a given connection *can* or *cannot* be established between a source and a destination which is H hops away when the multiplexing degree is K. In a trivial case where $K = 1$ (which is equivalent to having no time-multiplexing), the LM and the proposed PM approach are not different and will result in the same $P[H, 1]$. More specifically, since a connection can be established only if all the H external links along the path are available, we have

$$P_{LM}[H, 1] = P_{PM}[H, 1] = (1-u)^H \tag{5}$$

When the multiplexing degree $K > 1$, PM and LM approaches will result in different success probabilities. Consider first the case in which PM is used. We may begin with the probability that *none* of the K time slots is available on *all* the H links. Based on (5), this probability is

$$\overline{P}_{PM}[H, K] = \left(1 - P_{PM}[H, 1]\right)^K = \left(1 - (1-u)^H\right)^K$$

Since a connection can be established as long as there is one out of K time slots such that the same time slot is available along all the H links, we have

$$P_{PM}[H, K] = 1 - \overline{P}_{PM}[H, K] = 1 - \left(1 - (1-u)^H\right)^K \tag{6}$$

We now consider the case in which LM is used. For *a given link*, the probability that *none* of the K time slots is available is u^K, thus the probability that at least one time slot is available on that given link is $1 - u^K$. Since a connection can be established as long as on each of the H links along the path, one of the K time slots is available, we have

$$P_{LM}[H, K] = (1 - u^K)^H \tag{7}$$

To simplify our analysis, we assume that the effective packet generation rate r' is independent of time. Since P (which is either P_{PM} or P_{LM}) is the success probability, and v is the probability that a connection originating from a PE is established, we have

$$v = r' \cdot P \tag{8}$$

Since P is a function of u (see (6) or (7)), which in turn is related to v, we can combine (4) with (6) to get the following equation for PM,

$$r' \cdot [1 - (1 - (1-u)^H)^K] - 4u/H = 0 \tag{9}$$

Similarly, by combining (4) with (7), we get the following equation for LM,

$$r' \cdot \{(1 - u^K)^H - 4u/H = 0 \tag{10}$$

Given H, K, and r', we can solve (9) iteratively, using the

Secant method for example, to obtain the steady-state value of u for PM. Similarly, we can obtain the steady-state value of u for LM by solving (10).

In Fig. 9, we plot the solutions to (9) and (10) for specific H, K, and r'. As expected, the value of u increases with r'. The fact that for a given r', the value of u increases with K is consistent with previous results which show that, in general, multiplexing can improve both bandwidth utilization and network throughput [14], [15], [16]. Our results have also shown that for a given r' and K, u decreases with H. This is due to the fact that a longer path has a smaller chance of being established successfully.

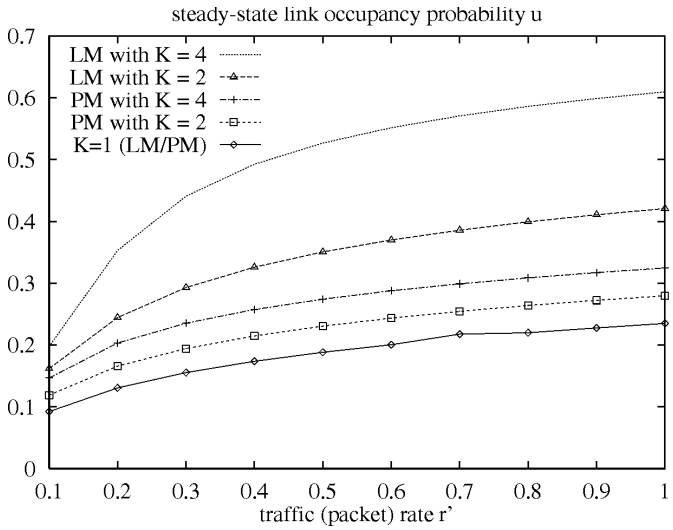


Fig. 9. Steady-state link occupancy probability u when $H = 8$.

After we obtain u, the success probability P can be obtained using (6) and (7) for PM and LM, respectively. Fig. 10 shows the steady-state success probability when $r' = 0.2$, which, similar to u, increases with K and decreases with H.

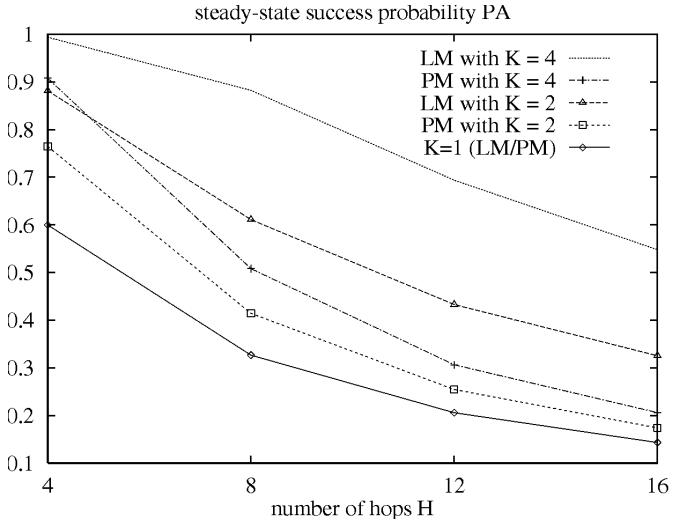


Fig. 10. Analytic results of steady-state success probability when $r' = 0.2$.

Having obtained the success probability P for either PM or LM, we can now determine the blocking time in the two approaches. If a request is satisfied the first time it is submitted, which occurs with probability P, the first packet can be sent after $K/2$ time slots on average. If t is the resubmission interval after each failure, then the connection will experience a latency (due to blocking) of $(t + K/2)$ time slots with probability $P(1 - P)$, and a latency of $(2t + K/2)$ time slots with probability $P(1 - P)^2$, and so on. Therefore,

$$L_B = P \cdot K / 2 + P(1-P)(t + K / 2) + P(1-P)^2(2t + K / 2) + \cdots$$
$$= P \times \sum_{i=0}^{\infty} (1-P)^i (i \times t + K / 2) \quad (11)$$

which results in

$$L_B = K / 2 + \frac{t \cdot (1-P)}{P} \quad (12)$$

Therefore, we may calculate the blocking time $L_B(PM)$ and $L_B(LM)$ by substituting P_{PM} and P_{LM} for P in the above equation. As a result, we can determine the communication latency $L(PM)$ and $L(LM)$ according to (1) and (2), respectively, as follows:

$$L(PM) = K / 2 + \frac{t \cdot \left(1 - P_{PM}[H, K]\right)}{P_{PM}[H, K]}$$
$$L(LM) = K / 2 + \frac{t \cdot \left(1 - P_{LM}[H, K]\right)}{P_{LM}[H, K]} + K \cdot (H - 1) \quad (13)$$

Finally, we can determine the improvement ratio I according to (3).

Assuming that $r' = 1.0$, $K = 4$, and $t = 4$ and 8, respectively, Table 1 shows some of the results obtained through the above probability analysis. From Table 1, one can see that in all cases, *PM* results in reduced communication latency when compared to LM. Note that since a larger t means a larger penalty for PM, the improvement ratio decreases with t. In addition, since the success probability of both PM and LM decreases with H *exponentially*, and the switching latency of LM alone increases with H *linearly*, H can affect the improvement ratios in both positive and negative ways. Our studies have found that in almost all the cases, I decreases with H as the negative effect dominates. These results are verified by our simulations, as will be discussed in the next subsection.

4.3 Simulation Study

In order to keep the above analysis simple, we have assumed that the probability of establishing a connection does not depend on the message length, nor on the message buffer size. In addition, the analysis assumes a node-symmetric network such as a torus. As a complementary part of our comparative study, simulations have also been conducted to evaluate the LM and PM approaches in a two dimensional $N \times N$ mesh.

The simulation program is written in C using the libraries from the YACSIM [32]. Statistical measures on the circuit-establishment latency resulted from PM and LM are collected with a confidence level no less than 90% and a confidence interval no larger than 0.1. In addition, average distances *in hops* between a source and a destination of all the connections established are also collected. These are used for the average values of H in (2), which, together with

TABLE 1
ANALYTIC RESULTS OF CONNECTION LATENCY AND IMPROVEMENT RATIO

$K = 4$	$t = 4$				$t = 8$			
H	2	4	6	8	2	4	6	8
$L(PM)$	2.88	7.80	14.53	22.64	3.76	13.60	27.07	43.29
$L(LM)$	6.37	16.75	27.82	39.12	6.75	19.51	33.64	48.25
I	54.8%	53.4%	47.7%	42.1%	44.3%	30.2%	19.5%	10.3%

the multiplexing degree K, can be used to determine $L(LM)$ and therefore the improvement ratio I. For example, we have simulated meshes with different sizes and found that the average value of H in an $N \times N$ network is about $0.66N$, which is consistent with the analytical result obtained with the following equation:

$$H = \frac{1}{N^4} \cdot \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N}\left(|i-k|+|j-l|\right) = \frac{2}{N^2} \cdot \sum_{i=1}^{N}\sum_{k=1}^{N}\left(|i-k|\right) \quad (14)$$

The simulation results presented in Figs. 11, 12, 13, 14, and 15 are obtained by limiting the message rate r not to exceed a certain point, beyond which the network will be saturated as indicated by a steady or even a decreasing network throughput. In practice, the actual traffic applied to the network would (and should) not exceed the saturation point. Our simulation results have indicated that within the range of parameter values we considered, the PM approach will always result in lower communication latency than the LM approach.

In Fig. 11, both improvement ratio I and communication latency in PM, $L(PM)$, are shown. These results are obtained using the following parameter values: the network size 100 ($N = 10$); the multiplexing degree $K = 4$; resubmission interval $t = 4$ time slots; each message contains $m = 2$ packets and each processor has a message buffer of size $b = 2$. Our simulation results indicated that the network is near saturation when message rate r reaches 0.3. At this point, the improvement ratio and communication latency in PM approach 60% and 13 time slots, respectively. At a low message rate (e.g., $r = 0.02$), the improvement ratio is almost 100%. This occurs because at a low message rate, $L(PM)$ is near 0 while $L(LM)$ is dominated by the switching latency, which is about 22 time slots. We also observe that the improvement ratio decreases with message rate. This is because, while the difference in the network propagation delay L_N between PM and LM remains almost constant (e.g., 22 time slots), the difference in the success probability between PM and LM increases with the network traffic load. Hence, the difference in the blocking time L_B between the two approaches increases with the network traffic load.

In Fig. 12, the effect of the buffer size, b on both the improvement ratio and the communication latency in PM is shown. It can be seen that in all cases, the improvement ratio is above 50%, although for a given message rate, the improvement ratio decreases with b. This is consistent with the previous observation because a larger value of b translates into a heavier traffic load for the same message rate.

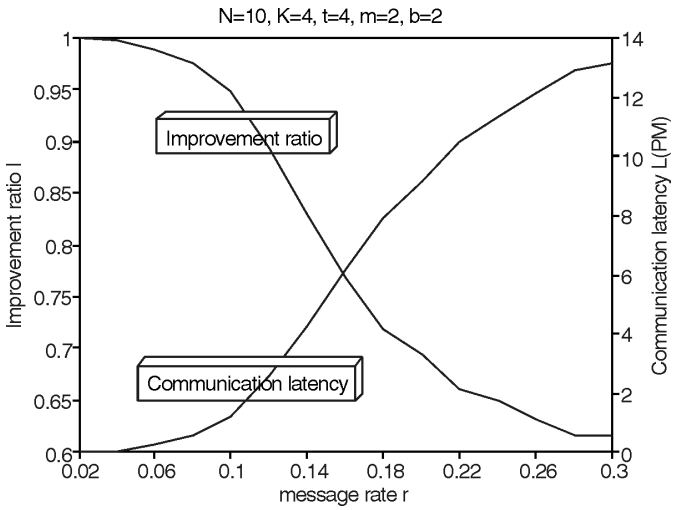


Fig. 11. Simulation results for different message rates.

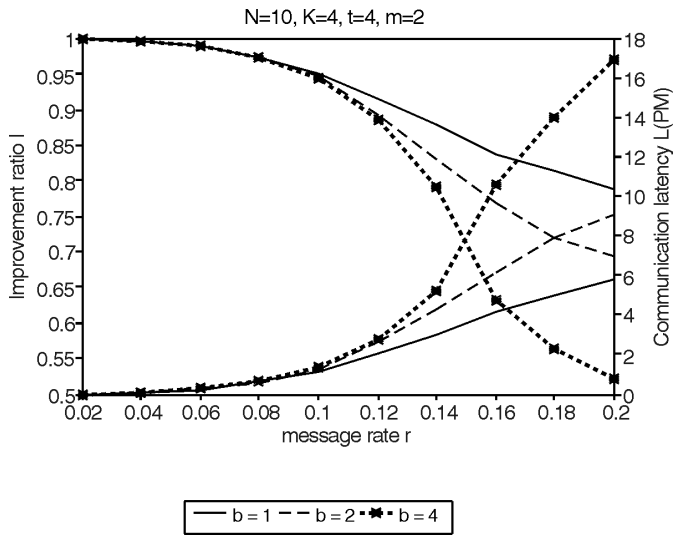


Fig. 12. Effect of buffer size b.

Fig. 13 shows that although the improvement ratio decreases with t, it is about 70% at $r = 0.14$ even for the case $t = 16$. This is due to the fact that one-time failure probability in establishing a connection is higher in PM than in LM (see (6) and (7)) and, thus, a larger t penalizes PM more than LM. Put differently, a larger t magnifies the disadvantage of PM relative to LM when establishing a connection (see (12)).

Fig. 14 shows that the improvement ratio increases with K. When $K = 1$, L(PM) is the same as L(LM), resulting in $I = 0$ (the horizontal line). Our simulation results have indicated that the network can tolerate a heavier traffic load with a larger K. In this regard, a larger K helps PM as much as LM in reducing the blocking time L_B. However, a larger K means a larger switching latency and hence a larger communication latency in LM. The result is a larger improvement ratio I.

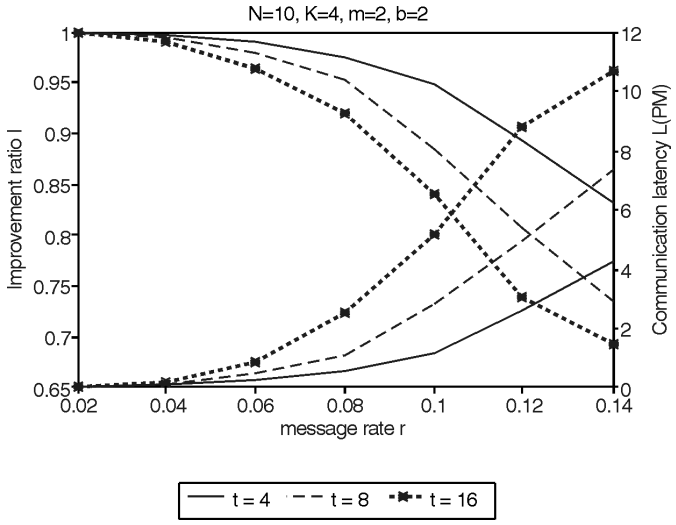


Fig. 13. Effect of resubmission interval *t*.

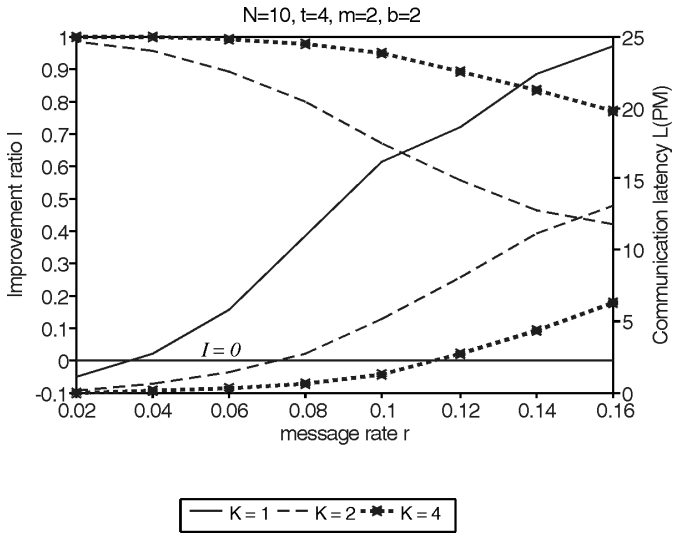


Fig. 14. Effect of multiplexing degree *K*.

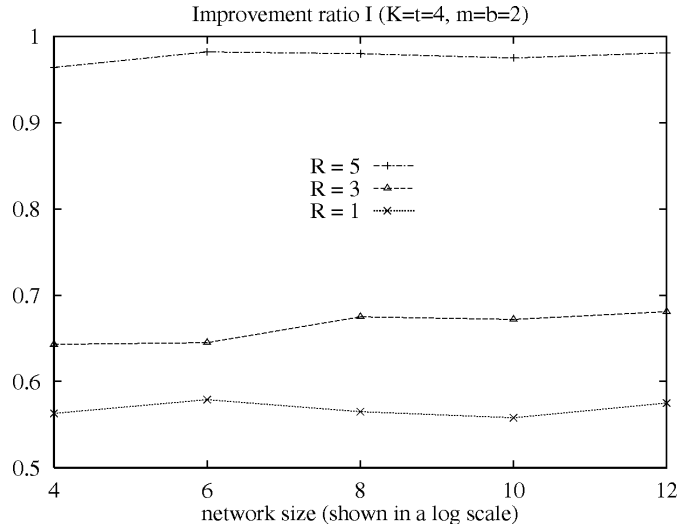


Fig. 15. Effect of network size (I versus $\log N^2$).

In order to investigate the effect of network size independently of the effect of traffic load, we compare $L(PM)$ and $L(LM)$ when the per-link load is the same for different values of N. One way to fix the per-link load is to use an r so that the average number of messages generated in a network is $R \cdot N$ for some R. Specifically, because each message travels an average distance on the order of N from its source to its destination, and the number of links in the network is of order N^2, then per link load is fixed if the total number of messages in the network is proportional to N. Note that, since there are N^2 PEs, each PE will generate a request with probability $r = RN / N^2$, or $r = R / N$.

Our simulation results have indicated that although the absolute value of the communication latency in both PM and LM increases with N^2, their difference also increases due to the increase in the average distance between a source and a destination H, which is linear in N. Therefore, as can be seen from Fig. 15, with a given value of R, the improvement ratio in networks of different sizes is about the same. It is worth noting that because of the finite buffer size b, having the same R in networks of different sizes can not guarantee that the *actual* load is also the same.

Finally, in order to investigate the effect of message length m independently of the traffic load, we obtain the improvement ratio resulted from the same r' but with different m and r. In addition, we have set the buffer size b so that the same number of packets (instead of messages) can be buffered. As can be seen from the results shown in Fig. 16, the improvement ratio increases slightly with m. This is because a larger m means that each connection is kept longer and a fewer connections need to be established to transfer the same number of packets. As a result, the negative effect of the blocking time of PM on the overall communication latency decreases. Therefore, PM gains more advantages over LM as m, which is proportional to the *temporal* locality in the communication, increases. Note that if the message size is large, then the message transmission time may dominate the communication time (see Section 4), making the reduction in the communication latency achieved in PM less important. However, in distributed shared-memory systems, message transfers are triggered by memory references, and the message size is usually small. In addition, since software latency in these message transfers is low, reducing hardware latency such as L_B and L_N via the use of PM becomes important.

To summarize, our comparative study has shown that PM is effective in reducing the communication latency in a wide range of practical situations. More specifically, as long as a network is not saturated with heavy traffic, an improvement ratio of at least 10% can be obtained. In particular, the improvement ratio increases with multiplexing degree and the temporal locality of communications (which is proportional to the message length), and scales with the network size. We note that in our simulations, uniform message routing (i.e., random destinations) is assumed. In many applications, spatial locality exists in communication patterns. For example, a processor may send a message to the processors within certain distance (i.e., number of hops) with a higher probability than to the other processors [33]. The effect of such nonuniform message routing on the latency reduction achievable in PM needs to be investigated further.

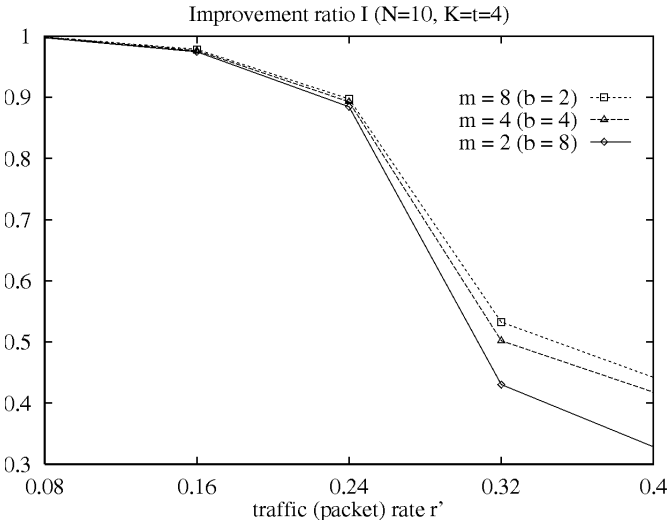


Fig. 16. Effect of message length (or connection duration) *m*.

5 Concluding Remarks

With the high bandwidth available in optics, it becomes increasingly important to reduce the communication latency which ultimately limits the performance of the multiprocessor systems that utilize optical interconnects. In this paper, we have proposed the *Path Multiplexing* (PM) approach as an alternative to the conventional *Link Multiplexing* (LM) approach for establishing connections in TDM networks. We have studied the issues related to the implementation of both approaches and compared the two in terms of the resulting communication latency through analysis and simulations.

The conventional LM approach uses switches that are capable of interchanging time slots while the PM approach does not. Although the LM approach may result in shorter blocking time due to its flexibility in selecting time slots for establishing a connection, we have found that the proposed PM approach can reduce the *overall* communication latency because it eliminates the delay caused by interchanging time slots. The results we have obtained on blocking probabilities of the two approaches in the TDM networks agree in principle with those of the two analogous approaches in the WDM networks [34], [35], [36], although in the latter, communication latency issues are quite different as interchanging wavelengths may incur little or no delays. We note that, aside from the issue of communication latency, the hardware and control complexity of a network using Time-Slot-Interchangers (TSIs) is higher than that using Time-Multiplexed Switches (TMS). In particular, networks using electronic TSIs are not transparent to data rate and format, and those using optical TSIs will have a fixed multiplexing degree and thus are incapable of adapting to different applications for which certain multiplexing degree yields optimal results [13], [14].

In a related work, it was found that PM can be nearly as effective as LM in realizing permutations in a class of networks including linear arrays, rings, meshes/tori, and hypercubes [37]. Further studies focusing on the cost-effectiveness of LM and PM on the network's capability of establishing random connections, and supporting general static or compiled communications are underway.

Acknowledgments

The authors would like to acknowledge partial support from the U.S. National Science Foundation under contract number MIP-9409864 and from the U.S. Air Force Office of Scientific Research under contract number AFOSR-89-0469. The authors would also like to thank the anonymous reviewers for their valuable comments which helped improve the quality of this paper. A preliminary version of the paper appeared in the Proceedings of the International Symposium on High-Performance Computer Architecture in January 1995.

References

[1] M. Feldman, S. Esener, C. Guest, and S. Lee, "Comparison Between Optical and Electrical Interconnects Based on Power and Speed Considerations," *Applied Optics*, vol. 27, pp. 1,742-1,751, May 1988.

[2] J. Goodman, F. Leonberger, S. Kung, and R. Athale, "Optical Interconnections for VLSI Systems," *IEEE Proc.*, vol. 72, pp. 850-866, 1984.

[3] P. Lalwaney, L. Zenou, A. Ganz, and I. Koren, "Optical Interconnects for Multiprocessors: Cost Performance Analysis," *Proc. Symp. Frontiers of Massively Parallel Computation*, pp. 278-285, Oct. 1992.

[4] R. Floren et al., "Optical Interconnects in the Touchstone Supercomputer Program," *SPIE Proc. Integrrated Optoelectronics for Comm. and Processing*, vol. 1,582, pp. 46-54, Oct. 1991.

[5] B. Kahle, E. Parish, T. Lane, and J. Quam, "Optical Interconnects for Interprocessor Communications in the Connection Machine," *Proc. IEEE Conf. Computer Design*, Oct. 1989.

[6] W. Dally and C. Seitz, "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks," *IEEE Trans. Computers*, vol. 36, no. 5, pp. 547-553, May 1987.

[7] C. Jesshope, P. Miller, and J. Yantchev, "High Performance Communications in Processor Networks," *Proc. 16th Int'l Symp. Computer Architecture*, pp. 150-157, 1988.

[8] L. Ni and P. McKinley, "A Survey of Wormhole Routing Techniques in Direct Networks," *Computer*, vol. 26, no. 2, pp. 62-76, Feb. 1992.

[9] Z. Guo et al., "Array Processors with Pipelined Optical Busses," *J. Parallel and Distributed Computing*, vol. 12, no. 3, pp. 269-282, 1991.

[10] H. Jordan, "Time Multiplexed Optical Computers," *Proc. of Supercomputing*, pp. 370–378, Nov. 1991.

[11] R. Melhem, D. Chiarulli, and S. Levitan, "Space Multiplexing of Waveguides in Optically Interconnected Multiprocessor Systems," *The Computer J.*, vol. 32, no. 4, pp. 362-369, 1989.

[12] C. Qiao and R. Melhem, "Time-Division Optical Communications in Multiprocessor Arrays," *IEEE Trans. Computers*, vol. 42, no. 5, pp. 577-590, May 1993.

[13] C. Qiao and R. Melhem, "Reconfiguration with Time-Division Multiplexed MINs for Multiprocessor Communications," *IEEE Trans. Parallel and Distributed Systems*, vol. 5, no. 4, pp. 337-352, Apr. 1994.

[14] C. Qiao, R. Melhem, D. Chiarulli, and S. Levitan, "Dynamic Reconfiguration of Optically Interconnected Networks with Time Division Multiplexing," *J. Parallel and Distributed Computing*, vol. 22, no. 8, pp. 268-278, Aug. 1994.

[15] P. Dowd, K. Bogineni, K.A. Aly, and J. Perreault, "Hierarchical Scalable Photonic Architectures for High Performance Processor Interconnection," *IEEE Trans. Computers*, vol. 42, no. 9, pp. 1,105-1,120, Sept. 1993.

[16] A. Ganz and Y. Gao, "A Time-Wavelength Assignment Algorithm for a WDM Star Network," *Proc. IEEE InfoCom*, pp. 2,144-2,150, 1992.

[17] D. Hunter and D. Smith, "New Architectures for Optical TDM Switching," *IEEE/OSA J. Lightwave Technology*, vol. 11, pp. 495-511, Mar. 1993.

[18] H. Jordan, D. Lee, K. Lee, and S. Ramana, "Serial Time Slot Interchangers and Optical Implementations," *IEEE Trans. Computers*, vol. 43, no. 11, pp. 1,309-1,318, Nov. 1994.

[19] R. Thompson, "Architectures with Improved Signal-to-Noise Ratio in Photonic Systems with Fiber-Loop Delay Lines," *IEEE J. Selected Areas in Comm.*, vol. 6, Aug. 1988.
[20] F. Cappello and C. Germain, "Towards High Communication Performance Through Compiled Communications on a Circuit Switched Interconnection Network," *Proc. Int'l Symp. High Performance Computer Architecture*, pp. 44-53, Jan. 1995.
[21] R. Cypher, A. Ho, S. Konstatinidou, and P. Messina, "Architectural Requirements of Parallel Scientific Applications with Explicit Communication," *Proc. Int'l Symp. Computer Architecture*, pp. 2-13, 1993.
[22] K. Alnes and B. Parady, "Gigabit SCI Cluster Interfaces," *Proc. Hot Interconnects III*, 1995.
[23] R. Gillett, "Memory Channel Network for PCI," *Proc. Hot Interconnects III*, 1995.
[24] S. Scott, "A New, Supercomputer-Class System Interconnect," *Proc. Hot Interconnects III*, 1995.
[25] D. Kiefer and V. Swanson, "Implementation of Optical Clock Distribution in a Supercomputer," Cray Research Inc., internal report, 1995.
[26] F. Leighton, *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes*. Morgan Kaufmann, 1992.
[27] S. Ramanan and H. Jordan, "Serial Array Shuffle-Exchange Architecture for Universal Permutation of Time Slots," *SPIE Proc., Digital Optical Computing II*, vol. 1,215, pp. 330-342, Jan. 1990.
[28] H. Shing and L. Ni, "A Conflict-Free Memory Design for Multiprocessor," *Proc. Supercomputing*, pp. 46-55, Nov. 1991.
[29] R. Thompson, R. Anderson, J. Camlet, and P. Giordano, "Experimental Modular Switching System with a Time-Multiplexed Photonic Center Stage," *Photonic Switching, OSA Technical Digest*, pp. 212-218, Mar. 1989.
[30] S. Nugent, "The iPSC/2 Direct-Connect Communications Technology," *Proc. Third Conf. Hypercube Concurrent Computers and Applications*, vol. 1, Jan. 1988.
[31] C.-L. Wu and M. Lee, "Performance Analysis of Multistage Interconnection Network Configurations and Operations," *IEEE Trans. Computers*, vol. 41, no. 1, pp. 18-27, Jan. 1992.
[32] J. Jump, *YACSIM Reference Manual, version 2.1*. Dept. of Electrical and Computer Eng., Rice Univ., Mar. 1993.
[33] D. Reed and R. Fujimoto, *Multicomputer Networks: Message-Based Parallel Processing*. MIT Press, 1987.
[34] I. Chlamtac, A. Ganz, and G. Karmi, "Lightpath Communications: An Approach to High-Bandwidth Optical WANs," *IEEE Trans. Comm.*, vol. 40, pp. 1,171-1,182, July 1992.
[35] K. Lee and V.O. Li, "A Wavelength-Convertible Optical Network," *IEEE/OSA J. Lightwave Technology*, vol. 11, no. 5/6, pp. 962-970, 1993.
[36] R. Ramaswami and K. Sivarajan, "Optimal Routing and Wavelength Assignment in All-Optical Networks," *Proc. IEEE Infocom*, pp. 970-979, June 1994.
[37] C. Qiao and Y. Mei, "On the Multiplexing Degree Required to Embed Permutations in a Class of Interconnection Networks," *Proc. IEEE Symp. High Performance Computer Architecture*, pp. 118-129, Feb. 1996.

Chunming Qiao received a BS in computer science and engineering from the University of Science and Technology of China in 1985, and a PhD in computer science from the University of Pittsburgh in 1993. He joined the Department of Electrical and Computer Engineering at the State University of New York at Buffalo as an assistant professor in September 1993 and received a National Science Foundation Research Initiation award in 1994. Dr. Qiao has published in the areas of multiprocessor interconnection networks, time-division and wavelength division multiplexed optical communications, and photonic switching. He helped organize and chaired session at the IEEE Conference on Military Communications (MILCOM '96) and SPIE's Conference on All-Optical Communication Systems in 1996 in addition to serving on the program committee of the International Conference on Massively Parallel Processing Using Optical Interconnection (MPPOI) in 1996. He is a member of the IEEE Computer Society and IEEE Communications Society.

Rami Melhem received a BE in electrical engineering from Cairo University in 1976, an MA in mathematics, and an MS in computer science from the University of Pittsburgh in 1981, and a PhD in computer science from the University of Pittsburgh in 1983. He is a professor of computer science at the University of Pittsburgh. He was previously an assistant professor at Purdue University and an assistant/associate professor at the University of Pittsburgh. He has published numerous papers in the areas of optical interconnections, fault tolerance, systolic architectures, and high performance computing. He served on program committees of several conferences and workshops and served as the general chair for the third International Conference on Massively Parallel Processing Using Optical Interconnections. He is a member of the editorial board of *IEEE Transactions on Computers*, and was a guest editor of a special issue of the *Journal of Parallel and Distributed Computing* on optical computing and interconnection systems. Dr. Melhem is a member of the IEEE Computer Society, the Association for Computing Machinery, and the International Society for Optical Engineering.