A Virtual Circuit Deflection Protocol

Emmanouel A. Varvarigos, *Member, IEEE,* and Jonathan P. Lang, *Student Member, IEEE*

***Abstract*— We propose a communication protocol, called the virtual circuit deflection (VCD) protocol, which combines some of the individual characteristics of virtual circuit switching and deflection routing. An advantage of the VCD protocol over previous (datagram) deflection schemes is that deflections in the former occur on a per session basis (or a per subsession basis, if sessions need to be split to find adequate capacity on the outgoing links), while in the latter, they occur on a per packet basis. This makes packet resequencing at the destination considerably easier to accomplish in the VCD protocol than in datagram deflection schemes. The VCD protocol exploits the storage arising from the high bandwidth-delay product of optical fibers to provide lossless communication with little buffering at the switches and without the need for advance reservations. This makes it particularly suitable for networks that use optical switching, where buffers are expensive to implement with current optical technology. We present a simple implementation of the VCD protocol for such networks, which requires only limited buffering, accomplished through the use of a minimal number of optical delay lines. We also analyze the performance of the protocol for the Manhattan Street network topology by using new analytical models. In particular, we examine the effect of the traffic load and the network size on the throughput and the length of the paths followed by the sessions, and compare the analytical results obtained with corresponding simulation results. Our results indicate that the VCD protocol is efficient under both light and heavy traffic conditions, especially when the link capacities are large compared to the basic rate of individual sessions, as is expected to be the case in future multigigabit networks.**

***Index Terms*— Deflection routing, Manhattan Street network, multigigabit networks, optical switching, performance analysis, tell-and-go protocol, virtual circuit switching.**

I. INTRODUCTION

TRANSMISSION rates of the order of 100 Gb/s or higher are currently feasible through the use of optical fiber technology and high-speed electronics. Having a communication link of that bit rate, however, does not necessarily result in a communication network of the same effective capacity. The development of efficient network control protocols, the quality of service they provide, and the buffering and processing requirements they impose on the switches are keys to the broad success of multigigabit networks. The main objectives in designing connection and flow-control protocols for multigigabit networks are to ensure lossless transmission, efficient utilization of the capacity, minimum pretransmission delay, small buffer requirements, and packet arrival in the correct order. These objectives, however, often appear to contradict each other. For example, when no advance reservations are made (so that pretransmission delay is minimized), deflection routing seems to be the only way to provide lossless communication without using substantial buffering and without underutilizing the network capacity; deflection schemes proposed to date, however, do not preserve the order of packets and are inconsistent with virtual circuit switching. The virtual circuit deflection (VCD) protocol that we propose and analyze in this paper attempts to simultaneously meet these objectives, with small hardware complexity.

The VCD protocol is a *virtual circuit switching* protocol of the *tell-and-go* variety, where data starts being transmitted shortly after the set-up packet of the session is sent. A preferred path is selected based on (possibly outdated) topology and link-utilization information available at the source at the time, and a set-up packet is sent on that path to establish a connection. The set-up packet is followed after a short delay, much shorter than the end-to-end round-trip delay required by wait-for-reservation types of protocols (e.g., [1], [2]) by the data packets, in this way avoiding the pretransmission delay associated with end-to-end reservations. If the capacity available at a preferred intermediate link is insufficient to accommodate the session, the set-up packet and the data packets that follow it may have to be routed over a different, longer path; we then say that the session is *deflected*. As we will see later, when the total outgoing capacity is equal to the total incoming capacity of a node, adequate capacity can always be made available on the outgoing links of an intermediate node to accommodate a new incoming session. This, however, may happen at the expense of interrupting (preempting) existing sessions that originate at that node, and/or splitting the new session into smaller subsessions, each of which follows a different path. The deflection or splitting of sessions at intermediate nodes is infrequent, and can happen only when the topology or link utilization information at the source is outdated and the network is congested.

An important advantage of the proposed VCD protocol over *datagram deflection schemes* (such as slotted [3] and unslotted [4] packet deflection schemes, and loop deflection schemes [5]) is that it significantly reduces the need for packet resequencing at the destination. This is because deflections in the former occur on a per session basis (or a per subsession basis, if session splitting is required to find adequate capacity on the outgoing links; see Section II), while in the latter, they occur on a per packet basis. Consequently, message reassembly at the destination, which is one of the main problems of deflection schemes [3], is easier with the VCD protocol. When a session is split, blocks of data have to be resequenced,

Manuscript received March 20, 1997; revised March 13, 1999; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor K. N. Sivarajan. This work was supported by the Defence Advanced Research Projects Agency (DARPA) under the MOST project.

The authors are with the Department of Electrical and Computer Engineering, University of California at Santa Barbara, Santa Barbara, CA 93106 USA (e-mail: manos@ece.ucsb.edu; jonathan@paros.ece.ucsb.edu).

Publisher Item Identifier S 1063-6692(99)05103-1.

instead of individual packets. This is important for multigigabit networks, where a session may involve the transfer of millions of packets. Moreover, in the VCD protocol, data packets are routed through a switch based on the virtual circuit identifier (or the virtual path identifier) they carry and the routing tables established by the set-up packet, maintaining in this way one of the main advantages of virtual circuit switching. By contrast, in the deflection protocols proposed to date, routing decisions are made individually for each data packet, each of which has to carry the destination address, making the switch processor a potential bottleneck of the design. A potential disadvantage of the VCD protocol (as well as other deflection schemes [3], [6]) is the fairness problem, which we discuss in Section III-A.

Traffic in high-speed networks can be switched either optically or electronically. Even though optical switching has advantages for circuit switching, it is generally considered incompatible with packet switching. This is because efficient packet switching requires substantial packet storage, which is difficult to provide with current optical technology (optical storage, using optical fiber loops with optical amplifiers and optical switches, is bulky and expensive compared to electronic storage). The VCD protocol provides lossless communication for data streams that are nearly uniform with small buffer space at the intermediate nodes. In Section IV, we present a particular implementation of the VCD protocol for networks using (almost all) optical switching, which employs a small number of optical delay lines to perform the buffering function. For other works that discuss optical implementations of deflection schemes we refer the reader to [4], [7]–[10].

Even though the effective utilization of idle links is an advantage, the increase of the number of links used per session as a result of deflections is a disadvantage of the VCD protocol. Thus, it becomes important to investigate the effects of the VCD protocol from both viewpoints. We analyze the performance of the VCD protocol for the Manhattan Street (MS) network topology under the assumption that all sessions have equal rates, and their source and destination nodes are uniformly distributed over all nodes of the network. We obtain results on the throughput, the average number of deflections, and other performance parameters of interest as a function of the traffic load, the network size, and the link capacities. Our analytical results are in close agreement with corresponding simulation results. Deflection routing protocols have previously been analyzed by several researchers, under a variety of assumptions on the underlying network topology [3], [5], [11]–[18]. Our model, analysis, and results are considerably different than those presented in previous works, where only packet-by-packet (datagram) deflections, instead of session (virtual circuit) deflections, were considered. As a result, session durations and rates played no role in these works, and packet arrivals at a node and their destinations could be assumed to be independent. This is very different from our model, where we focus on sessions (virtual circuits) rather than packets (datagrams), and the previous assumptions are no longer valid. The VCD protocol cannot be considered as a special case of the unslotted deflection scheme with cut-through routing proposed in [4], because in the VCD protocol, multiple sessions can share the same link, and a node does not have to store the packets of a whole session (as may be the case with the natural extension of unslotted schemes, where a packet of variable length is taken to the limit where it represents a whole session). Our analysis does not assume nodes to have any global information about the utilization of the network links, other than for their own outgoing links. Since, in practice, such information will be available and a source will make an effort to select a path of sufficient unused capacity to route a new session, the performance of the VCD protocol in a real network is expected to be better than that predicted by our analysis.

The organization of the remainder of the paper is as follows. In Section II, we introduce the VCD protocol and describe how it can be combined with appropriate queueing disciplines to provide lossless communication. In Section III, we compare the VCD protocol with wait-for-reservation and tell-and-go type of protocols for gigabit networks, and discuss issues related to fairness. In Section IV, we describe an implementation of the protocol for networks using optical switching. In Section V, we evaluate the performance of the VCD protocol for the MS network topology. In Section VI, we present and discuss the analytical and simulation results obtained. Finally, in Section VII, we conclude the paper.

II. The VCD Protocol

The objectives that we set for the new protocol are:

1) lossless communication;
2) little buffer requirements;
3) small pretransmission delay;
4) efficient utilization of the capacity;
5) virtual circuit switching;
6) simple or no resequencing at the destination.

In this section, we describe the VCD protocol and show how it can be combined with other techniques to meet its objectives.

Call requests (sessions) are assumed to arrive at a source with a specified destination, and bandwidth requirement. A path with adequate residual capacity is then computed at the source based on (possibly outdated) topology and link utilization information that may available at the source at that time (the performance analysis to be given in Sections V and VI, however, assumes that a node has information only about its outgoing links). After determining a route through the network, a set-up packet is transmitted over the path to set the routing tables and reserve capacity at intermediate nodes, followed after a short delay by the data packets. If the set-up packet is successful in reserving capacity on all of the links on the path to the destination, the VCD protocol looks like the usual reservation protocols, with the difference that the reservation (set-up) phase and the transmission phase overlap in time (see also the discussion in Section III and Fig. 3, therein). If the residual capacity on a link is not sufficient to accommodate the new session, the session may have to be deflected and/or split into smaller subsessions, as described below.

We focus on a particular intermediate (i.e., nonsource, nondestination) node, where a set-up packet arrives requesting rate r. We let R_t, R_f, and R_i to be the total capacity occupied

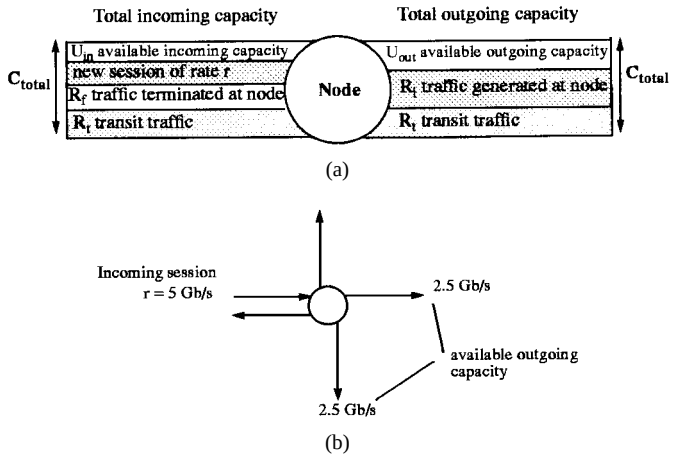


Fig. 1. (a) We illustrate the capacity occupied by existing sessions on the incoming and outgoing links of a node when the set-up packet of a new session arrives. The total (over all links) outgoing capacity is assumed to be equal to the total incoming capacity of the node. If the available outgoing capacity at a node is not sufficient to serve a new transit session of rate r, it may be necessary to preempt a session originating at that node in order to free some capacity. Such sessions resume transmission when the session that interrupted them ends. (b) The available outgoing capacity and/or the capacity that may become available through the preemption of existing sessions originating at the node may not all belong to the same outgoing link. In that case, the new incoming session may have to be split into two or more subsessions of total rate r that are routed over different outgoing links.

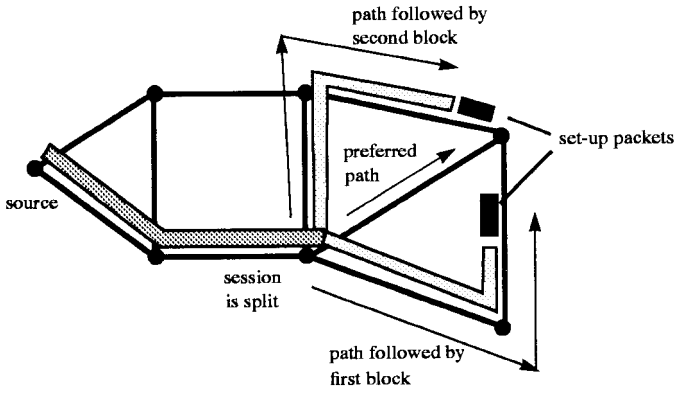


Fig. 2. We illustrate the situation where a session has to be split into two different subsessions, because the capacity available on a single link is not sufficient to accommodate it. In this example, both subsessions are deflected because no capacity was available on the preferred link. The case where one subsession is routed over the preferred link, while the other(s) is (are) deflected, is also possible.

by transit, terminating, and initiating sessions in progress at that node, respectively, when the set-up packet arrives. We also let U_{out} and U_{in} [Fig. 1(a)] be the total unused capacity on the outgoing and incoming links of the node, respectively. Note that when the set-up packet arrives at the node, it has already reserved capacity equal to r on the link on which it arrives. Since the total incoming capacity is equal to the total outgoing capacity of a node, we have

$$C_{\text{total}} = r + R_t + U_{\text{in}} + R_f = R_t + U_{\text{out}} + R_i$$

which implies

$$U_{\text{out}} + R_i \geq R_f + r \geq r. \tag{1}$$

Therefore, a set-up packet that arrives at an intermediate node requesting rate r can always find capacity equal to r to reserve on the outgoing links of the node. This may, however, require the interruption (preemption) of one or more of the existing sessions that initiate at that node (by releasing the capacity of such sessions, total outgoing capacity up to $U_{\text{out}}+R_i$ becomes available, which by (1) is sufficient to accommodate the new session). When a session is preempted, network resources at the source and along the path are released to accommodate the new session. It is possible that the outgoing capacity that is available, or that may become available through the preemption of existing sessions originating at a node, may not all belong to the same outgoing link of the node. In that case, the session may have to be split into two or more subsessions of smaller rates (Figs. 1(b) and 2), each of which is routed over a different path to the destination. Sessions that are interrupted may resume transmission when the session that preempted them ends (either because it is completed, or because a control packet is sent to its source requesting it to pause). When a session is split into a total of, e.g., k subsessions, packets belonging to different subsessions may arrive at the destination out of order; packets, however, belonging to the same subsession will always arrive in the correct order. Resequencing k blocks of packets (each of which is ordered) at the destination is much easier to accomplish than resequencing individual packets. This is one of the main advantages of the VCD protocol over other deflection protocols, where packets are deflected independently of each other, and the order of the packets in a session may be completely destroyed. Data packets in the VCD protocol are routed based on their virtual circuit identifier (VCI), using the lookup tables established by the set-up packet.

It is possible for sessions to be deflected such that the paths contain loops. This may arise after a series of deflections, or if a set-up packet is deflected immediately to the previously visited node. In either case, the bandwidth reserved in the loop is inefficiently used and it is desirable to remove the loop. However, unless the set-up packet visits the intermediate node for the second time prior to the arrival of the first data packet, it is unclear whether the added protocol complexity associated with removing the loop outways the efficiency benefits.

Allowing sessions to follow very long paths can waste network resources, increasing the probability that future sessions will be blocked or forced to take even longer paths. To avoid the waste that occurs when a session follows a very long path due to deflections, we may request that a session is dropped when the set-up packet has traveled more than H hops without reaching its destination. The parameter H can be chosen to be equal to a multiple (e.g., two or three times) of the shortest distance between the source and the destination of the session, and it may also be dependent on the current congestion in the network. A session that has undergone too many deflections is dropped by transmitting a control packet to the source, requesting it to cease transmitting new packets. Data packets sent prior to the arrival of the control packet at the source can either be dropped or allowed to remain in the network until they reach their destination (possibly over a very long path), while the remaining data is sent by the source later, over a different path.

It is possible, in the VCD protocol, for a session to preempt itself. This situation may arise when a set-up packet arrives, after undergoing several deflections, back at its source, and has no other way to obtain the required outgoing capacity. When a session preempts itself, its source stops transmitting new packets, while data packets that have been sent in the meantime circulate in a loop. When this happens, the source may decide to drop these packets immediately, or it may let them circulate until the set-up packet has traveled a total of H hops, in the hope that a way out of the loop will be found (e.g., some capacity leading out of the loop may become available). Our simulations indicate that self preemptions happen infrequently.

The time gap between the transmission of the set-up packet and the transmission of the first data packet from a source is chosen to be equal to the maximum number of hops H allowed for the particular session times the processing time of a set-up packet at a node. In other words, the gap must be at least as large as the minimum time by which the connection set-up phase and the data transmission phase should be separated to ensure that data packets do not overpass the set-up packet. For networks using optical switching, this delay should be large enough to permit the electronic processing of the set-up packet, without it being overpassed by the data packets, which will mostly remain in the optical domain (except for their VCI, which will probably have to be processed electronically).

The VCD protocol is designed to provide lossless communication for sessions that have constant rate, or sessions that have certain smoothness properties (to be discussed shortly), or sessions that have variable rate but can tolerate the delay induced when transforming them into smooth sessions through the use of input flow control. Constant-rate sessions can clearly be switched with little buffer space at the nodes. If more burstiness is allowed then additional buffer space, which depends on the degree of burstiness, is required to provide lossless communication. Following the discussion in [28], we view the time axis on a link as being divided into frames of length equal to T slots, where a slot is equal to the transmission time of a packet. A session is said to have the (R,T)-*smoothness property* at a node if at most R packets $(R \in \{1,\cdots,T\})$ of the session arrive at that node during a frame. By using a leaky bucket scheme [29] to shape traffic at the source, and the stop-and-go queueing discipline [28] to forward traffic at intermediate nodes, a session can be made to have the (R,T)-smoothness property throughout the network. Stop-and-go queueing requires buffer space for at most $3T$ packets per link, is consistent with the FIFO queueing discipline, and can be implemented with little processing overhead by artificially delaying the packets at a switch. Therefore, if the VCD protocol is combined with stop-and-go queueing, and buffer space equal to $3T$ packets per link is available at the nodes, lossless communication is guaranteed for all sessions that have the (R,T)-smoothness property. The larger T is, the more bursty the session is allowed to be, and the larger the required buffer space is. A disadvantage of stop-and-go queueing is that link capacity can be allocated to a session only at discrete levels that are multiples of C/T, where C is the link capacity. This reduces the flexibility in assigning rates to sessions, but it also poses an upper bound on the number of subsessions in which a session may be split, and, therefore, on the maximum number of packet blocks that may have to be resequenced at the destination. In particular, if the maximum length of a path is equal to H hops and the node degree is equal to d, then a session of rate RC/T may be split into at most $\min(R, Hd)$ subsessions, in the worst case. For example, a session of minimal rate C/T will never have to be split.

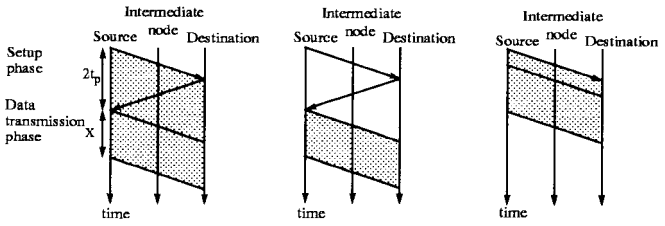


Fig. 3. (a) In WRVC protocols, a set-up packet is sent to the destination to reserve the required capacity and set the routing tables at the intermediate nodes. During the set-up phase, the capacity that is (explicitly) reserved for a session remains idle and cannot be used by other sessions. This is inefficient because this capacity is actually needed at least one round-trip delay after the arrival of the set-up packet at the node. In a typical WRVC protocol, the capacity is blocked for $X + 2t_p$, where X is the duration of the session and t_p is the end-to-end propagation delay. (b) In the ERVC protocol, capacity is blocked for other sessions only for the holding time X. However, the ERVC protocol still requires a round-trip propagation delay for the set-up phase. (c) In tell-and-go protocols, such as the VCD protocol, capacity is occupied for time X, plus the time offset between the transmission of the set-up packet and the first data packet of the session. The VCD protocol does not require an end-to-end round-trip delay for connection set-up, minimizing in this way the pretransmission delay and increasing the network capacity utilization.

III. Comparison With Other Lossless Protocols for Gigabit Networks

A sizable portion of the traffic in future gigabit networks will involve the high-speed transfer of massive amounts of data at nearly constant rates, and will require guaranteed lossless delivery and explicit reservation of bandwidth (e.g., the constant-bit rate class of asynchronous tranfer mode (ATM) traffic). Most of the connection control protocols designed to deal with this type of traffic use explicit reservations of link capacity prior to the transmission of any data. Since a source has to wait for an acknowledgment from the destination in such protocols before it can transmit any data packets, we refer to them as *Wait-for-Reservation Virtual Circuit* (WRVC) protocols. As illustrated in Fig. 3(a), WRVC protocols tend to be inefficient in terms of link utilization, since capacity is reserved for more time than a session requires. Furthermore, the pretransmission delay required for the set-up phase is often significant compared to the delay requirements of the session, and unwarranted if the network load is light. Even with a fast reservation protocol [1], the set-up phase requires at least time equal to a round-trip end-to-end propagation delay to complete. For gigabit networks this delay is substantial (of the order of 40 ms for coast-to-coast communication) compared to the transmission time of a packet (of the order of 10 ns for ATM cells and 40-Gb/s links) and the holding time of a session (10 ms for one million packets and 40-Gb/s links). A protocol designed to overcome the inefficiency problem of WRVC

protocols is the *Efficient Reservation Virtual Circuit* (ERVC) protocol, proposed in [19], which uses information about the session holding times and permits advanced reservations in order to fully exploit the available bandwidth. Capacity is reserved for a session starting from the time at which it is needed, and for a duration equal to the holding time of the session. However, the ERVC protocol also requires a pretransmission delay equal to at least an end-to-end round-trip propagation delay.

For sessions in which the round-trip pretransmission delay is not acceptable, *tell-and-go* protocols are more appropriate. In such protocols, the set-up packet is followed after a short delay by the data packets, achieving in this way a pipelining between the set-up phase and the data transmission phase, and reducing the pretransmission delay to the minimum possible. If the unused capacity found by the set-up packet at an intermediate node is not adequate to accommodate the session, the excess data packets are usually buffered at the node, and backpressure [20] is exercised to upstream nodes to control the source transmission rate. For transmission rates of the order of 100 Gb/s and 100-km-long links, the buffer space required by node-to-node backpressure protocols is of the order of 100 Mb/link. At such speeds, buffers of adequate size are difficult to build, and their design is not flexible enough to accommodate the requirements of backpressure protocols (e.g., they have to be first in first out (FIFO) buffers shared by many sessions, which does not permit the use of per session queueing as required by most backpressure protocols; see [21] and [22] for some difficulties that arise with such buffer designs).

The VCD protocol avoids the difficulties associated with wait-for-reservation and backpressure-based protocols, and can ensure lossless communication with little buffering and a small pretransmission delay. Since link capacity is reserved for duration slightly larger than the holding time of a session, and is available for the remaining time, it has an efficiency advantage over WRVC protocols [Fig. 3(c)]. This is particularly important for high-speed networks where propagation times are often large compared to the typical holding time of sessions. Also, the small buffer requirements of the VCD protocol make it more suitable than backpressure-based protocols for very high-speed networks, and all-optical networks in particular. In the VCD protocol, if the set-up packet is successful in finding adequate capacity on the first link on its path, it will be able to establish an end-to-end connection (unless it preempts itself). This provides quick feedback information to the source on the success or failure of a connection, avoiding the wasteful repetition of the set-up phase and the blocking of resources that arise when a set-up packet successfully reserves part of its route and releases it later due to the unavailability of capacity at subsequent nodes.

A. Fairness Issues

A potential disadvantage of the VCD protocol (and deflection schemes in general [3]) is that it can cause the network to operate unfairly, especially under heavy load conditions. This is because continuing sessions have priority over originating sessions. An upstream node that has plenty of available outgoing capacity can take all the capacity that it requires, while the node that follows takes what is left over. Moreover, sessions originating at downstream nodes may have to be preempted by sessions originating at upstream nodes. Fig. 4 illustrates a situation where a source is locked out with the VCD protocol.

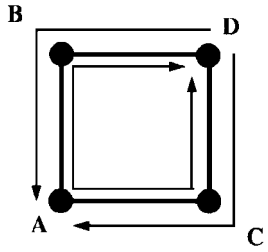


Fig. 4. Assume, for simplicity, that all link capacities and session rates are equal to one unit. Node A has two connections to node D, and node D has two connections to node A, taking all the capacity and preventing nodes B and C from accessing the network. If the connection from node A via node B to node D is terminated, then node B will be able to establish a connection to node D, but it will again be preempted and blocked out if node A resumes transmitting to node D.

If sessions select their preferred paths based on link utilization information available at their source at the time, they will avoid using heavily loaded paths where preemptions or deflections are likely to occur. This is only a partial solution, however, since the information available at a node about distant links may be outdated, especially when the round-trip delays in the network are comparable to the session holding times. To reduce the fairness problem, sources that are preempted may send a throttle packet to the source of the session that preempted them, requesting it to cease transmission. Finally, nodes that have enough available outgoing capacity may help other nodes that are not treated fairly get access to the network by establishing (upon request) "dummy connections" to such nodes. The destination of a dummy connection will then be able to originate a connection with rate at least equal to the rate of the dummy connection (a node is locked out only when all of its capacity is taken by transit traffic, and can always originate a session with rate equal to the total rate of the sessions that terminate at that node; see Fig. 1). This idea is similar to the notion of "token packets" used in datagram deflection networks to alleviate the fairness problem [6].

IV. Implementation of the VCD Protocol for All-Optical Networks Using Optical Delay Lines

Networks using optical switching offer the potential of larger transmission speeds than networks using electronic switching by eliminating the need for optical to electronic to optical (O/E/O) conversion of the transmitted data signal at intermediate switches, the so-called *electronic bottleneck*. (For packet switching, O/E/O conversion may still have to be done for the header, though, creating another potential bottleneck; see [7], [9], [10], [14], [23], and [24] for possible solutions.) The difficulty in implementing buffers of large size with current optical technology has led many researchers to argue that circuit switching is more appropriate than packet

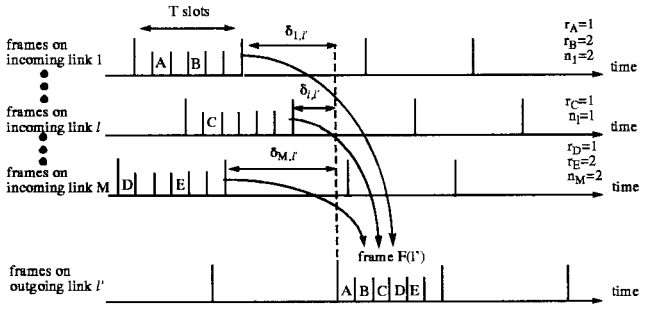


Fig. 5. We illustrate the incoming and outgoing frames at a node. Packets not intended for outgoing link l' are not shown. Packets intended for link l' are assigned to outgoing slots according to the packing rule described in the text.

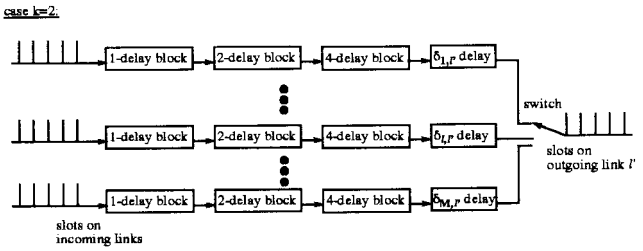


Fig. 6. We illustrate the output system for a particular outgoing link l'. Each delay block can be implemented by a switch and an optical fiber of appropriate length, as shown in Fig. 7. Depending on the distance between the arriving and the departing slot of a packet, the state of each delay block is set so that a packet is delayed until its assigned outgoing slot comes. The $\delta_{l,l'}$ delays are implemented using fibers of appropriate length, and account for the misalignment between incoming and outgoing frames. Clearly, even though the frames on the outgoing links of a node can be synchronized, if desired, this is unrealistic to assume for the incoming links of the node, since it would require global synchronization and exact knowledge of the lengths of the links connecting different nodes.

switching [25] for networks using optical switching. Since the VCD protocol requires little buffering at the switches, it is natural to examine its suitability for all-optical networks. In this section, we give an implementation of the VCD protocol for networks using optical switching, where the buffering function is performed through the use of a small number of optical delay lines. Our design uses some of the ideas developed by Lin and Gallager [26], but its hardware complexity is about half of that in [26]. (This reduction in complexity comes from the flexibility permitted by the VCD protocol in assigning outgoing slots to packets, and the use of a special assignment rule). The simplicity of the implementation suggests that the VCD protocol is an interesting alternative to circuit switching, at least for applications where the end-to-end round-trip delay is large compared to the delay requirements and the holding times of the sessions.

In our design, the time axis on the incoming and outgoing links of a node is divided into frames, each of which has duration equal to T packet slots. All sessions using a link are assigned transmission rates that are integer multiples of C/T, where C is the link capacity. Thus, a session with transmission rate equal to iC/T, $i = 1,2,\cdots,T$, can transmit up to i packets in a frame. The larger the value of parameter T, the greater the flexibility we have in assigning rates to sessions (but also, as will see, the larger the hardware complexity of the switch). The frames on the incoming and the outgoing links of a node will not, in general, be synchronized. We let $\delta_{l,l'}$ be the phase difference between the beginning of the frames on links l and l'. To preserve frame integrity, we request that packets arriving in frame $F(l)$ of incoming link l and destined for outgoing link l' are transmitted in the first frame $F(l')$ of link l' that starts after the end of $F(l)$ (Fig. 5).

We let M be the number of incoming links of a node, and n_l, $l = 1,2,\cdots,M$, $n_l \in \{0,1,\cdots,T\}$, be the number of packets that arrive during frame $F(l)$ and are transmitted during the corresponding frame $F(l')$ of link l'. The VCD protocol guarantees that $\sum_{l=1}^{M} n_l \leq T$, and therefore, there are always enough slots in outgoing frame $F(l')$ to serve all packets that have to be transmitted in it. For this to happen, however, it is necessary to delay a packet arriving in incoming frame $F(l)$ until the time of its transmission on outgoing frame $F(l')$. The required delay can take any value between 1 and $2T-1$ slots, and it can be implemented using $2T-1$ optical delay lines of variable lengths between 1 and $2T-1$ slots, for a total fiber length per incoming link equivalent to $T(2T-1)$ slots. For link capacities of the order of 50 Gb/s and ATM cells, the slot duration is approximately 10 ns, and each kilometer of fiber can store about 500 cells. For 250-ms frames, we have $T = 25\,000$, and the total length of the fiber per link needed for storage is $T(2T-1)/500 = 2.5\times 10^6$ km, which is clearly excessive. For a design using delay lines to be practical, the number of delay elements has to be reduced. In what follows, we give a construction that uses only $\log T$ (as opposed to $2T-1$) delay elements per link, with a total fiber length proportional to T (as opposed to $2T^2$).

The delay lines that implement the buffering system for a particular outgoing link l' are depicted in Fig. 6 for the case $T = 2^k = 4$. A 2^i-delay block at stage i can be in state 0 or 1. If the block is in state 0, it does not introduce any delay, while if it is in state 1, it introduces delay equal to 2^i slots. In the VCD protocol, a packet may have to be delayed by anywhere between 1 and $2T-1 = 2^{k+1}-1$ slots. Clearly, all delays in this range can be implemented by appropriately choosing the states of the delay blocks. Since different packets have to be delayed by different amounts, the state of a block will, in general, change at the end of a slot. However, as long as the arrival pattern on the incoming links remains the same (for example, if the packets of each session arrive periodically in the incoming frames and while no new sessions are added), the sequence of states used will be the same for successive frames.

For the design given in Fig. 6 to work, it is necessary that two different packets never appear during the same slot at the output of a stage (see also Fig. 7). To prevent such collisions, the assignment of incoming slots (packets) to outgoing slots cannot be arbitrary. In what follows, we present an assignment method, called the *packing rule*, which guarantees that no collisions arise in the system of Fig. 6. We focus on a particular frame $F(l')$ of an outgoing link l'. Consider a packet A that arrives in slot $x_A \in \{0,1,\cdots,T-1\}$ of frame $F(l)$, and assume that it is the r_A^{th} packet destined for outgoing link l' to arrive in $F(l)$ (the integer r_A, $r_A \in \{1,\cdots,n_l\}$, will be referred to as the *rank* of packet A). Then, according to the packing rule, packet A is assigned to slot $y_A = \sum_{i=1}^{l-1} n_i + r_A - 1$, $y_A \in \{0,1,\cdots,T-1\}$, of the outgoing

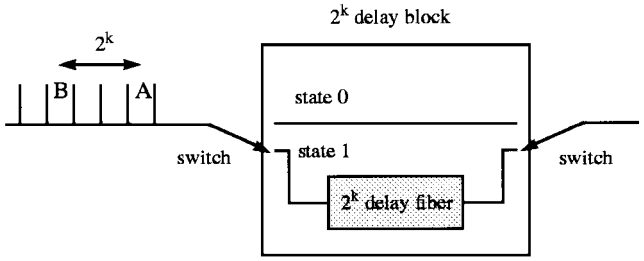


Fig. 7. We illustrate the design of a 2^k-delay block. A packet collision may occur in the case where packet B lags packet A by 2^k slots, and packet A passes through the upper branch (state 0), while packet B passes through the lower branch (state 1) of the 2^k-delay block. We prove in Theorem 1 that if the packing rule is used to assign packets to outgoing slots, two packets will never appear at the output of a stage during the same slot.

frame $F(l')$ (Fig. 5). As shown in the following theorem, when packets are assigned to outgoing slots according to the packing rule, no collisions occur at the outputs of the delay blocks.

Theorem 1: When the packing rule is followed, two packets will never appear at the output of a stage during the same slot.

Proof: Clearly, packets arriving on different incoming links will never collide since they are routed through different delay lines, and they are assigned to different outgoing slots. Consider two packets A and B that arrive on incoming link l during slots x_A and x_B of the same frame $F(l)$, and they both have to be transmitted in frame $F(l')$ of outgoing link l'. We let r_A and r_B be their ranks, and we assume (without loss of generality) that $r_A < r_B$. According to the packing rule, packets A and B are assigned to outgoing slots

$$y_A = \sum_{i=1}^{l-1} n_i + r_A - 1 \tag{2}$$

$$y_B = \sum_{i=1}^{l-1} n_i + r_B - 1 \tag{3}$$

respectively. For packets A and B to collide at the output of stage 0, they should arrive at successive slots (that is, we should have $x_B = x_A + 1$, which implies $r_B = r_A + 1$ and $y_B = y_A + 1$), and A should be delayed by one slot, while B should not be delayed at all at stage 0. This cannot happen because A and B have been assigned to successive slots in outgoing frame $F(l')$, which implies that they have to be delayed by the same amount, and therefore their delay at stage 0 must be the same. Thus, packets A and B cannot collide at the output of stage 0.

We now generalize the previous argument to show that packets A and B cannot collide at the output of any stage $i > 0$. To show this, we let $(x_0^A, x_1^A, \cdots, x_{k-1}^A)$ and $(y_0^A, y_1^A, \cdots, y_{k-1}^A)$ be the binary representations of x_A and y_A, and $(x_0^B, x_1^B, \cdots, x_{k-1}^B)$ and $(y_0^B, y_1^B, \cdots, y_{k-1}^B)$ be the binary representations of x_B and y_B, respectively. We also let T_A^i [or T_B^i] be the slot, counting from the beginning of frame $F(l)$, at which A (or B) is transmitted at the output of stage i, and we let $(t_0^{A,i}, t_1^{A,i}, \cdots, t_k^{A,i})$ (or $(t_0^{B,i}, t_1^{B,i}, \cdots, t_k^{B,i})$, respectively) be its binary representation. Since the delay introduced at any subsequent stage j, $j > i$, is either 0 or 2^j, we have that $(t_0^{A,j}, t_1^{A,j}, \cdots, t_i^{A,j}) = (y_0^A, y_1^A, \cdots, y_i^A)$ for all $j > i$. In other words, at any stage after stage i, the i least significant bits of the slot in which a packet appears are identical to the i least significant bits of the outgoing slot to which the packet has been assigned and will finally appear. (For example, for $i = 0$ we have $t_0^{A,j} = y_0^A$, for all $j > 0$, which means that after stage 0, packet A will always appear at the output of a stage in an even slot, if its assigned outgoing slot y_A is even, and in an odd slot, if y_A is odd.) For packets A and B to appear during the same output slot of stage i, we should have that

$$(t_0^{A,i}, t_1^{A,i}, \cdots, t_k^{A,i}) = (t_0^{B,i}, t_1^{B,i}, \cdots, t_k^{B,i})$$

which can happen only if

$$\begin{aligned}(y_0^A, y_1^A, \cdots, y_i^A) &= (t_0^{A,i}, t_1^{A,i}, \cdots, t_i^{A,i}) \\ &= (t_0^{B,i}, t_1^{B,i}, \cdots, t_i^{B,i}) \\ &= (y_0^B, y_1^B, \cdots, y_i^B).\end{aligned}$$

In other words, for packets A and B to collide at the output of stage i, the i least significant bits of y_A and y_B should be identical. This implies (since $y_A < y_B$) that $y_B \geq y_A + 2^{i+1}$, and [in view of (2) and (3)], $r_B \geq r_A + 2^{i+1}$. By the definition of the rank, we then have that $x_B \geq x_A + 2^{i+1}$, which means that there should be a delay of at least 2^{i+1} slots between the arrivals of A and B in the incoming frame $F(l)$. Since the first $i-1$ stages reduce this delay by at most $1 + 2 + \cdots + 2^{i-1} = 2^i - 1$ slots, the slots at which packets A and B appear at the input of stage i will be separated by a distance of at least $2^{i+1} - 2^i + 1 = 2^i + 1$ slots. Since stage i cannot introduce delay larger than 2^i slots, it follows that packets A and B will not collide at the output of stage i. □

V. Analysis for the MS Network

Beginning with this section, we turn our attention to the performance analysis of the VCD protocol. We will assume that the underlying topology is the MS network, which is a two-connected regular mesh network with unidirectional communication links. The reason we focus on the MS network is that it is a natural topology for gigabit networks, since it can cover a large geographical area with small total fiber length. Also, because of its regularity and symmetry properties, the MS network has been analyzed extensively in the literature for datagram deflection schemes (see, for example, [4], [6]–[8], [11], [15], [16], [27]). We believe that the results obtained are characteristic of the performance of the VCD protocol for other topologies that offer, as the MS network does, a large number of alternative paths between any source–destination pair of nodes.

The $X \times Y$-dimensional wraparound mesh consists of $N = XY$ processors arranged along the points of a two-dimensional (2-D) space that have integer coordinates. There are X processors along the x-dimension and Y processors along the y-dimension, where X and Y are even numbers. Each processor has two outgoing links, one horizontal and one vertical. The horizontal links are directed eastwards on even rows and westwards on odd rows, while the vertical links are directed northwards on even columns and southwards on odd columns. Each processor is represented by a pair (x, y), with $0 \leq x \leq X - 1$ and $0 \leq y \leq Y - 1$.

A. Performance Analysis for a Single Class of Users

In this section, we analyze the performance of the VCD protocol when the topology is a square MS network, with $X = Y = \sqrt{N}$ nodes along each dimension. We assume that external session (connection) requests are generated at each node over an infinite time horizon according to a Poisson process of rate λ, and their destinations are uniformly distributed over all nodes of the network. All sessions have rate equal to one unit, and their holding times are independent and exponentially distributed with mean $1/\mu$. The capacity of each link is taken to be equal to m units. The units by which link capacity and transmission rates are measured is immaterial and is left unspecified. Therefore, m can be viewed as the number of sessions that can simultaneously use a link and is equal to T if the frame structure of Section IV is used; in multigigabit networks, m is expected to be a very large number. We also assume that the time required to process a set-up packet at a node is small, so that the time offset between the transmission of the set-up packet and the data packets is negligible, compared to the average holding time of the session. (Alternatively, we assume that the processing time of the set-up packet is included in the session's holding time.)

Since the session rates are equal to one unit and the uncommitted capacity on a link is always an integer number of units, sessions do not have to be split, and all packets of a session arrive at their destination in the correct order. It is still possible, however, for sessions to be deflected and/or preempted.

A session using a given link l is called an *originating session* if l is the first link on the session's path, and a *transit session* if l is an intermediate link. We similarly distinguish two types of set-up packets: *originating set-up packets*, which are emitted by the source node of a session, and *transit set-up packets*, which are emitted by intermediate nodes on the session's path. A session that reaches its destination over link l is called a *terminating session* for link l, and a set-up packet that reaches its destination is called a *terminating set-up packet*. When both of the outgoing links of a node lie on a shortest path to the destination, the node is called a *don't care* node for that destination; otherwise, it is called a *preference* node. Upon its generation at a source or upon its arrival at an intermediate node, a set-up packet selects a preferred link according to the following rule.

Persistent Rule: If the current node is a "don't care" node, one of the links is chosen with equal probability as the preferred one. If the current node is a "preference" node, the preferred link is the one that lies on the shortest path.

A transit set-up packet attempts to reserve capacity on its preferred link, preempting if necessary a session originating at that link. If this is not possible, the session is routed over the other link of the node, preempting, if necessary, some session originating on that link. An originating session is accepted only if there is capacity available on its preferred link to accommodate it; that is, sessions are never deflected on their first hop. An originating session that is not accepted is said to be blocked, and must try to establish a connection at a later time. A session that is preempted attempts again to establish a circuit after a random delay, and if it succeeds, the transmission of data continues from the point at which the session was interrupted. Since session holding times are assumed to be exponentially distributed, the remaining holding time of a session that has been preempted is again exponentially distributed with the same parameter. Sessions that are preempted or blocked are randomly mixed back into the input queues so that the combined process of exogenous and retrial set-up attempts can be approximated by a Poisson process.

We focus on sessions with destination $(0,0)$, and let $\bar{D}(i,j)$ [or $D(i,j)$] be the average number of additional links that will be used by a transit (or originating, respectively) set-up packet currently located at node (i,j), whose destination is node $(0,0)$. We let p be the probability that an arriving transit set-up packet fails to reserve the required capacity on its preferred outgoing link (therefore, such a set-up packet is deflected if the current node is a preference node). We then have

$$\bar{D}(i,j) = 1 + \begin{cases} \frac{1}{2}[\bar{D}(i_1,j_1) + \bar{D}(i_2,j_2)], \\ \quad \text{if } (i,j) \text{ is a don't care node} \\ (1-p)\bar{D}(i_1,j_1) + p\bar{D}(i_2,j_2), \\ \quad \text{if } (i,j) \text{ is a preference node and} \\ \quad (i_1,j_1) \text{ is the preferred next node} \end{cases} \tag{4}$$

and

$$D(i,j) = 1 + \begin{cases} \frac{1}{2}[\bar{D}(i_1,j_1) + \bar{D}(i_2,j_2)], \\ \quad \text{if } (i,j) \text{ is a don't care node} \\ \bar{D}(i_1,j_1), \\ \quad \text{if } (i,j) \text{ is a preference node and} \\ \quad (i_1,j_1) \text{ is the preferred next node} \end{cases} \tag{5}$$

where (i_1,j_1) and (i_2,j_2) are the outgoing neighbors of (i,j). Also, we clearly have $D(0,0) = \bar{D}(0,0) = 0$. In writing (4) and (5), we have taken into account that sessions cannot be deflected in their first hop. If the deflection probability p is known, the preceding equations can be applied iteratively on the MS network to calculate $D(i,j)$ and $\bar{D}(i,j)$ for all nodes (i,j). The total average number of links used by a session can then be obtained as

$$D = \frac{1}{N-1} \sum_{(i,j)\neq(0,0)} D(i,j). \tag{6}$$

In what follows, we present an analytical method for calculating the deflection probability p.

We distinguish between two types of transit set-up packets arriving at a node. Transit set-up packets that arrive on a horizontal (or vertical) link and select according to the persistent rule the horizontal (or vertical) link as their preferred outgoing link are called *straight-through* set-up packets. Transit set-up packets that arrive on a horizontal (or vertical) link and select according to the persistent rule the vertical (or horizontal) link as their preferred outgoing link are called *bend* set-up packets. We let $\bar{\Theta}(i,j)$ be the average number of additional nodes at which a transit set-up packet currently at node (i,j) will have a straight-through horizontal preference until it reaches its destination node $(0,0)$. Using the symmetry of the MS

network, we have

$$\bar{\Theta}(i,j) = \begin{cases} \frac{1}{2}[1 + \bar{\Theta}(i_1,j_1) + \bar{\Theta}(j_2,i_2)], \\ \quad \text{if } (i,j) \text{ is a don't care node} \\ 1 + (1-p)\bar{\Theta}(i_1,j_1) + p\bar{\Theta}(j_2,i_2), \\ \quad \text{if } (i_1,j_1) \text{ is the preferred next node} \\ p\bar{\Theta}(i_1,j_1) + (1-p)\bar{\Theta}(j_2,i_2), \\ \quad \text{if } (i_2,j_2) \text{ is the preferred next node} \end{cases} \tag{7}$$

where (i_1,j_1) and (i_2,j_2) are the horizontal and vertical neighbors of (i,j), respectively. Also, we clearly have $\bar{\Theta}(0,0) = 0$. If the deflection probability p is known, the preceding equation can be applied iteratively on the MS network to calculate $\bar{\Theta}(i,j)$ for all nodes (i,j). The average probability of a straight-through preference can then be obtained as

$$\Theta = \frac{1}{(N-1)(D-1)} \sum_{(i,j)\neq(0,0)} \Theta(i,j)$$

where

$$\Theta(i,j) = \begin{cases} \frac{1}{2}[\bar{\Theta}(i_1,j_1) + \bar{\Theta}(j_2,i_2)], \\ \quad \text{if } (i,j) \text{ is a don't care node} \\ \bar{\Theta}(i_1,j_1), \\ \quad \text{if } (i_1,j_1) \text{ is the preferred next node} \\ \bar{\Theta}(j_2,i_2), \\ \quad \text{if } (i_2,j_2) \text{ is the preferred next node} \end{cases} \tag{8}$$

where (i_1,j_1) and (i_2,j_2) are the horizontal and vertical neighbors of (i,j), respectively.

We denote by B the probability that an originating session (either new or reattempting due to blocking or preemption) is blocked, and by E the probability that a session is interrupted (preempted) before it is completed. We assume that the retransmissions of sessions that are blocked or preempted are sufficiently randomized so that the total arrival rate of originating sessions requesting a particular outgoing link of a node is a Poisson process with rate

$$\lambda_1^* = \frac{\lambda}{2(1-B)(1-E)}. \tag{9}$$

The factor $1/2$ in the preceding expression accounts for the probability that a session selects one of the two outgoing links of its source for its first hop. Since the average number of intermediate links (excluding the first link) used by a session is equal to $D-1$, the average rate with which transit set-up packets are emitted on a link is equal to

$$\lambda_2 = \lambda_1^*(1-B)(D-1) = \frac{\lambda(D-1)}{2(1-E)}. \tag{10}$$

Also, the average rate with which terminating set-up packets arrive at a node is $\lambda_3 = \lambda_1$.

We say that a node is in state

$$\bar{X} = (X_a, X_b, X_c, X_d, X_{ab}, X_{ad}, X_{cb}, X_{cd})$$

if there are X_a (or X_c) sessions terminating over its horizontal (vertical, respectively) incoming link, X_b (or X_d) sessions originating on its horizontal (vertical, respectively) outgoing link, X_{ab} (or X_{cd}) transit sessions arriving over the horizontal (or vertical) incoming link and leaving over the horizontal (or vertical, respectively) outgoing link, and X_{ad} (or X_{cb}) transit

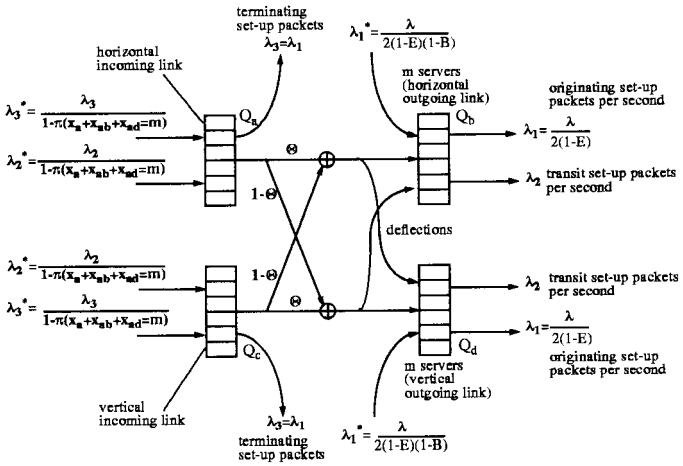


Fig. 8. The auxiliary system $\hat{\mathcal{Q}}$ has four groups of servers, each having m servers. There is no waiting space in the system. Originating, transit, and terminating customers arrive according to a Poisson process with input rates λ_1^*, λ_2^*, and λ_3^*, respectively. Transit customers have a straight-through preference with probability Θ, and have preemptive priority over originating customers, as explained in the text. Originating customers, if admitted, use a server for an exponential amount of time with parameter μ, or until they are preempted. Transit and terminating customers, if admitted, use a server for an exponential amount of time with parameter $\mu + \epsilon$.

sessions arriving over the horizontal (or vertical) incoming link and leaving over the vertical (or horizontal, respectively) outgoing link (Fig. 8). The set of feasible states is

$$\mathcal{F} = \{\bar{X} \geq 0 \mid X_a + X_{ab} + X_{ad} \leq m, X_c + X_{cb} + X_{cd} \leq m, \\ X_b + X_{ab} + X_{cb} \leq m, X_d + X_{ad} + X_{cd} \leq m\}.$$

We also let $\pi(\bar{X})$ be the steady-state probability that a node is in state $\bar{X}$.

We will approximate $\pi(\bar{X})$ as the stationary distribution of an auxiliary system $\hat{\mathcal{Q}}$, defined as follows (see also Fig. 8). The system $\hat{\mathcal{Q}}$ has four groups of servers (labeled Q_a, Q_b, Q_c, and Q_d), each of which has m identical servers and no waiting space. The groups Q_a and Q_c will be referred to as *incoming* groups, while the groups Q_b and Q_d will be referred to as *outgoing* groups of servers. We also refer to groups Q_a and Q_b as groups of the *top level*, and to groups Q_c and Q_d as groups of the *bottom level*. There are three types of customers, to be referred to as *originating*, *transit*, and *terminating* customers. Originating customers arrive at each outgoing group of servers (Q_b or Q_d) according to a Poisson process with rate λ_1^*. Transit and terminating customers arrive at each incoming group of servers (Q_a or Q_c) according to a Poisson process with rates λ_2^* and λ_3^*, respectively. Originating, transit, or terminating customers that find all servers in the group at which they arrive busy are dropped, never to appear again. An originating or terminating customer that is not dropped obtains one server in the group at which he arrives. A transit customer that arrives in an incoming group (Q_a or Q_c) obtains one server in the group of servers at which he arrives, and obtains an additional server in one of the outgoing groups (Q_b or Q_d), in the following way. The transit customer selects one of the outgoing groups as its preferred outgoing group. The preferred outgoing group is with probability Θ the outgoing group that is at the same level (top or bottom) with the incoming group at which he

arrived, and with probability $1-\Theta$ it is the outgoing group that is at the other level from the incoming group at which he arrived. The transit customer then tries to obtain a server in its preferred outgoing group (Q_b or Q_d). Transit customers have preemptive priority over originating customers in Q_b and Q_d. That is, a transit customer that finds its preferred outgoing group of servers busy, can preempt an originating customer in that group. If all servers in that group are busy serving transit customers, it tries to obtain a server in the nonpreferred group, preempting if necessary an originating session in that group. Once a transit customer is accepted in Q_a or Q_c, it is guaranteed to always find a server in Q_b or Q_d, given the above preemption rule. Originating customers that are accepted in the system use a server for an exponential amount of time with mean $1/\mu$, unless they are preempted before the completion of their service. Transit and terminating customers that are accepted in the system leave the system after an exponential amount of time with mean $1/(\mu+\epsilon)$, where the parameter ϵ is taken to be the "probabilistic rate" at which a transit session is preempted due to arrivals of set-up packets at its source, and will be defined precisely later. We also ask that the rate λ_2 at which transit set-up packets are emitted on a link of the MS network is the same with the rate at which transit customers are accepted in system $\hat{\mathcal{Q}}$, and the rate at which terminating packets are received at an incoming link of the MS network is the same with the rate at which terminating customers are accepted in system $\hat{\mathcal{Q}}$. For this to hold, we should have $\lambda_2^*=\lambda_2/G$ and $\lambda_3^*=\lambda_3/G$, where

$$G=1-\Pr(\bar{X}: X_a+X_{ab}+X_{ad}=m).$$

The probability that a session attempting to establish a connection is blocked at its first hop is

$$B=\sum_{\bar{X}:X_b+X_{ab}+X_{cb}=m}\pi(\bar{X}). \tag{11}$$

We define the *deflection probability* p as the probability that a transit set-up packet fails to reserve capacity on its preferred link (such a packet is deflected if the current node is a preference node). To determine p, we focus on a transit set-up packet arriving over the horizontal link (the case where a set-up packet arrives over a vertical link is symmetric). When such a packet arrives, the node cannot be in a state $\bar{X}$ with $X_a+X_{ab}+X_{ad}=m$, because the total incoming rate at that horizontal link (excluding the new session) has to be less than m. We assume that an arriving transit set-up packet finds a node in a typical state, except for states where $X_a+X_{ab}+X_{ad}=m$. Under this approximating assumption, the probability p that a transit set-up packet fails to reserve capacity on its preferred link is given by

$$p=\sum_{\substack{\bar{X}:X_{ab}+X_{cb}=m\\ X_a+X_{ab}+X_{ad}\neq m}}\Theta\pi(\bar{X})/G + \sum_{\substack{\bar{X}:\ X_{ad}+X_{cd}=m\\ X_a+X_{ab}+X_{ad}\neq m}}(1-\Theta)\pi(\bar{X})/G. \tag{12}$$

The first (or second) term in (12) accounts for the case where the set-up packet has a straight-through (or bend, respectively) preference and finds all the capacity in its preferred outgoing link occupied by transit sessions. To find the probabilistic rate ϵ at which a particular transit session $\mathcal{S}$ is preempted due to arrivals of set-up packets at its source, we focus at the source node of $\mathcal{S}$ and assume (without loss of generality) that the session's first hop is over a horizontal link. We then have

$$\epsilon=\lambda_2^*\Bigg[\sum_{\substack{\bar{X}:X_b+X_{ab}+X_{cb}=m\\ X_a+X_{ab}+X_{ad}\neq m\\ X_b\neq 0}}\frac{\Theta}{X_b}\frac{\pi(\bar{X})}{\Pr(\bar{X}:X_b\neq 0)} + \sum_{\substack{\bar{X}:X_b+X_{ab}+X_{cb}=m\\ X_a+X_{ab}+X_{ad}\neq m\\ X_b\neq 0\\ X_{ad}+X_{cd}=m}}\frac{(1-\Theta)}{X_b}\frac{\pi(\bar{X})}{\Pr(\bar{X}:X_b\neq 0)} + \sum_{\substack{\bar{X}:X_b+X_{ab}+X_{cb}=m\\ X_c+X_{cb}+X_{cd}\neq m\\ X_b\neq 0}}\frac{(1-\Theta)}{X_b}\frac{\pi(\bar{X})}{\Pr(\bar{X}:X_b\neq 0)} + \sum_{\substack{\bar{X}:X_b+X_{ab}+X_{cb}=m\\ X_c+X_{cb}+X_{cd}\neq m\\ X_b\neq 0\\ X_{ad}+X_{cd}=m}}\frac{\Theta}{X_b}\frac{\pi(\bar{X})}{\Pr(\bar{X}:X_b\neq 0)}\Bigg]. \tag{13}$$

The first term in the product of (13) is the effective rate λ_2^* at which transit set-up packets arrive at the source of $\mathcal{S}$ over an incoming link. The term within the brackets accounts for the probability with which set-up packets preempt session $\mathcal{S}$. In particular, the first (or second) term in the summation accounts for preemptions of $\mathcal{S}$ by transit set-up packets arriving over the horizontal link with a straight-through (or bend, respectively) preference. The third (or fourth) term in the summation accounts for preemptions of $\mathcal{S}$ by transit set-up packets arriving over the vertical link with a bend (or straight-through, respectively) preference. Each term in the summation within the brackets is equal to the probability of a particular (straight-through or bend) preference, multiplied with the probability that the set-up packet finds the node at a state where a session in $\mathcal{Q}_b$ has to be preempted (conditional on that $X_b\neq 0$, since we know that $\mathcal{S}$ originates at $\mathcal{Q}_b$), and multiplied with the probability $1/X_b$ that $\mathcal{S}$ is the particular session in $\mathcal{Q}_b$ (among the X_b originating sessions) to be preempted.

We now calculate the probability E that a session $\mathcal{S}$ that has been accepted is preempted before it is completed. While $\mathcal{S}$ is in service, transit set-up packets arriving at its source preempt it with rate ϵ. Since the rate at which S is preempted is ϵ and the rate at which it is normally terminated is μ, E can be approximated as

$$E=\frac{\epsilon}{\epsilon+\mu}. \tag{14}$$

When a session that is preempted reestablishes a connection and resumes service, it does so from the point at which it

was interrupted, and its remaining time is still exponentially distributed with mean $1/\mu$.

To calculate the steady-state probabilities $\pi(\bar{X})$ for all states $\bar{X} \in \mathcal{F}$, we write down the global balance equations of the Markov chain that corresponds to the auxiliary system $\hat{\mathcal{Q}}$. If the parameters λ_1^*, λ_2^*, λ_3^*, and ϵ are known, then the global balance equations together with the equation

$$\sum_{\bar{X} \in \mathcal{F}} \pi(\bar{X}) = 1 \tag{15}$$

give the steady-state probabilities. These parameters, however, depend on the values of the steady-state probabilities. Equations (5)–(15), together with the global balance equations, give a system of equations that can be solved by using the method of successive approximations. The following section presents the results that we obtained.

VI. Analytical and Simulation Results

In this section, we present our results on the throughput, the average path length, and other performance parameters of interest for the VCD protocol in an MS network topology. These results were obtained by solving numerically the analytical expressions given in Section V, and using simulations.

To verify the accuracy of our analysis, we modeled the network using a discrete event simulator written in $C{+}{+}$, and we compare our analytical results with the corresponding simulation results. In the simulation, new sessions are generated and placed in an event queue (ordered in time) such that the arrival rate λ of new sessions to each node is Poisson distributed, and the session durations are exponentially distributed with unit mean $(1/\mu = 1)$. Sessions that are blocked or preempted are randomly assigned a new arrival time (according to a Poisson distribution with rate $\lambda/10$) and inserted back into the event queue. This mechanism is used so that the combined process of exogenous and retrial set-up attempts can still be approximated by a Poisson process. Since the time offset between the transmission of the set-up packet and the data packets is assumed to be negligible compared to the average holding time of the session (see Section V-A), the set-up phase is instantaneous in the simulation. To calculate the statistics, we averaged the data within bins, where each bin corresponds to 50 000 sessions terminating normally; we discarded the data from the first bin to remove the transient effects. For each simulation point, data was collected and averaged in five bins (or 250 000 terminated sessions); we found the average statistics between bins had converged to within 1%.

A natural measure of the performance of the VCD protocol is the *inefficiency ratio* $\eta(\lambda)$, defined as the ratio

$$\eta(\lambda) = \frac{D(\lambda)}{D(0)} \tag{16}$$

of the average path length $D(\lambda)$ taken by a session for a given arrival rate λ, over the average shortest-path length $D(0)$ of the MS network topology. The inefficiency ratio characterizes the effectiveness with which the VCD protocol uses the network capacity for a given network load. In Fig. 9, we illustrate $\eta(\lambda)$ as a function of the external arrival rate λ (measured in sessions per node per unit of time), for a 6×6 MS network, and link capacities $m = 1$ and $m = 2$. We also illustrate the deflection probability p at a preference node, the preemption probability E, and the blocking probability B. We define the *stable* region as the operating region where the connection request queue remains finite; stability is not directly related to B, and it is possible to have B considerably less than one and still be in the unstable region. The results in Fig. 10, and the analytical model of Section V, assume that the network is operating in the stable region, and that all sessions generated are eventually served (possibly after being blocked or preempted and reattempting several times). This was confirmed by our simulations which showed that an average of 50 000 sessions were generated and completed per bin in the stable region. Note that as the external arrival rate λ increases, the preemption probability E is the first of these probabilities to approach one. We let $\lambda = \lambda_{\max}(m)$ be the maximum stable external arrival rate for a given link capacity m. As shown in Fig. 9, increasing the capacity m by a factor of two (from $m = 1$ to $m = 2$) increases the maximum throughput $\lambda_{\max}(m)$ by more than a factor of two, indicating that the larger the link capacity m, the more efficient is the operation of the VCD protocol.

To obtain a necessary condition for stability, note that external session requests are generated in the network at a total rate of λN sessions per unit of time, and each of them uses on the average $D(\lambda)$ links for an average duration equal to $1/\mu$. Since the total network capacity is $2Nm$, a necessary condition for stability is

$$\frac{\lambda N D(\lambda)}{\mu} \leq 2Nm$$

or equivalently

$$\eta(\lambda) \leq \frac{2m\mu}{\lambda D(0)}. \tag{17}$$

In order to investigate the behavior of the network in the unstable region, where the external arrival rate λ per node is larger than what the network can sustain, we rely on simulations. Fig. 10 illustrates simulation results for the various parameters of interest for values of λ in the stable and unstable regions. The horizontal axis is the normalized arrival rate per unit of capacity λ/m, so that the curves corresponding to different values of m can appear on the same plot. Note that the preemption probability E increases monotonically with λ, approaching one, while the parameters p, $\eta(\lambda) = D(\lambda)/D(0)$, and B increase with λ, but eventually reach a plateau. The network load at which the plateau is reached coincides with the load at which the preemption probability E becomes close to one, and the network is in the unstable region. This indicates that at heavy traffic load, session preemptions act as a "built-in" flow-control mechanism that prevents the deflection probability and the average path length from increasing beyond some point, and limits the blocking probability by freeing capacity that can be used by new sessions. The increase in efficiency of the VCD protocol when m increases is evident from the lower values that p and $\eta(\lambda)$ take when m is large.

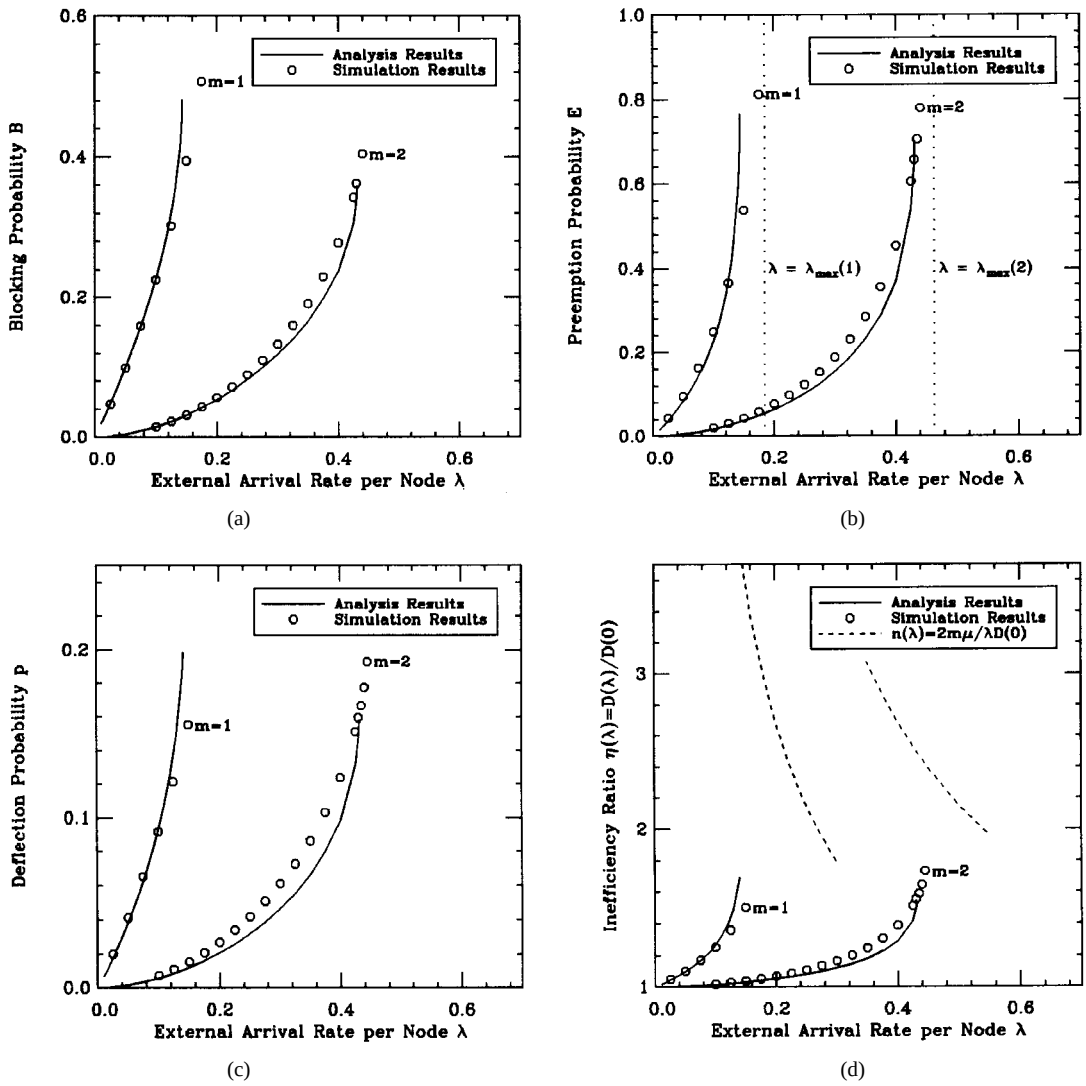


Fig. 9. We illustrate the blocking probability B, the preemption probability E, the deflection probability p, and the inefficiency ratio $\eta(\lambda)$, as a function of the external arrival rate per node λ, for a 6×6 MS network with capacities $m = 1$ and $m = 2$.

For example, for $m = 20$, the deflection probability p is always less than 0.015 [Fig. 10(c)] and the lengths of the paths taken are on the average within 5% from the shortest path length [Fig. 10(d)], for any value of the external arrival rate λ. Increasing m also increases the number of delay lines required for buffering (this increase goes as $\log m$ if the implementation of Section IV is used).

Since a preempted session continues from the point from which it was interrupted when its transmission resumes, there is no penalty for preempting a session (except for the fairness problem); a waste, however, of resources occurs whenever a session is deflected. This explains the efficiency of the VCD protocol at heavy loads, where E approaches one. The dashed line in Fig. 10(d) illustrates the necessary condition for stability as given by (17): points to the right of this curve correspond to unstable operation, that is, to session departure rates that are smaller than the external arrival rates. In the unstable region, originating sessions will preempt existing sessions to find adequate capacity for transmission. These newly accepted sessions will in turn quickly be preempted by arriving sessions (either new or reattempting), as the cycle continues. This behavior of cyclic preemption in the unstable region will delay the completion of sessions, as sessions will have to retry many times before completing normally, and some sessions may never be completed. Note that for large m, the load $\lambda_{\max}(m)$ at which the VCD protocol becomes unstable is very close to the limits of the stability region [as given by the necessary condition of (17)].

Borgonovo *et al.* [4] used simulations to analyze an unslotted deflection scheme with cut-through routing. The results that we obtained for the case $m = 1$ are similar to the results obtained in [4] for $\tau \approx 0$ (note that the results in [4] are given in terms of throughput versus offered traffic, and blocked packets do not retry to enter the network). This was expected, because if we view a packet of variable length in [4] as representing a whole session (this corresponds to $\tau \approx 0$ in [4]), and we assume that all sessions have rates equal to the link capacity, so that sharing of links by multiple sessions cannot happen (this corresponds to $m = 1$ in the VCD protocol), the efficiency with which the two protocols

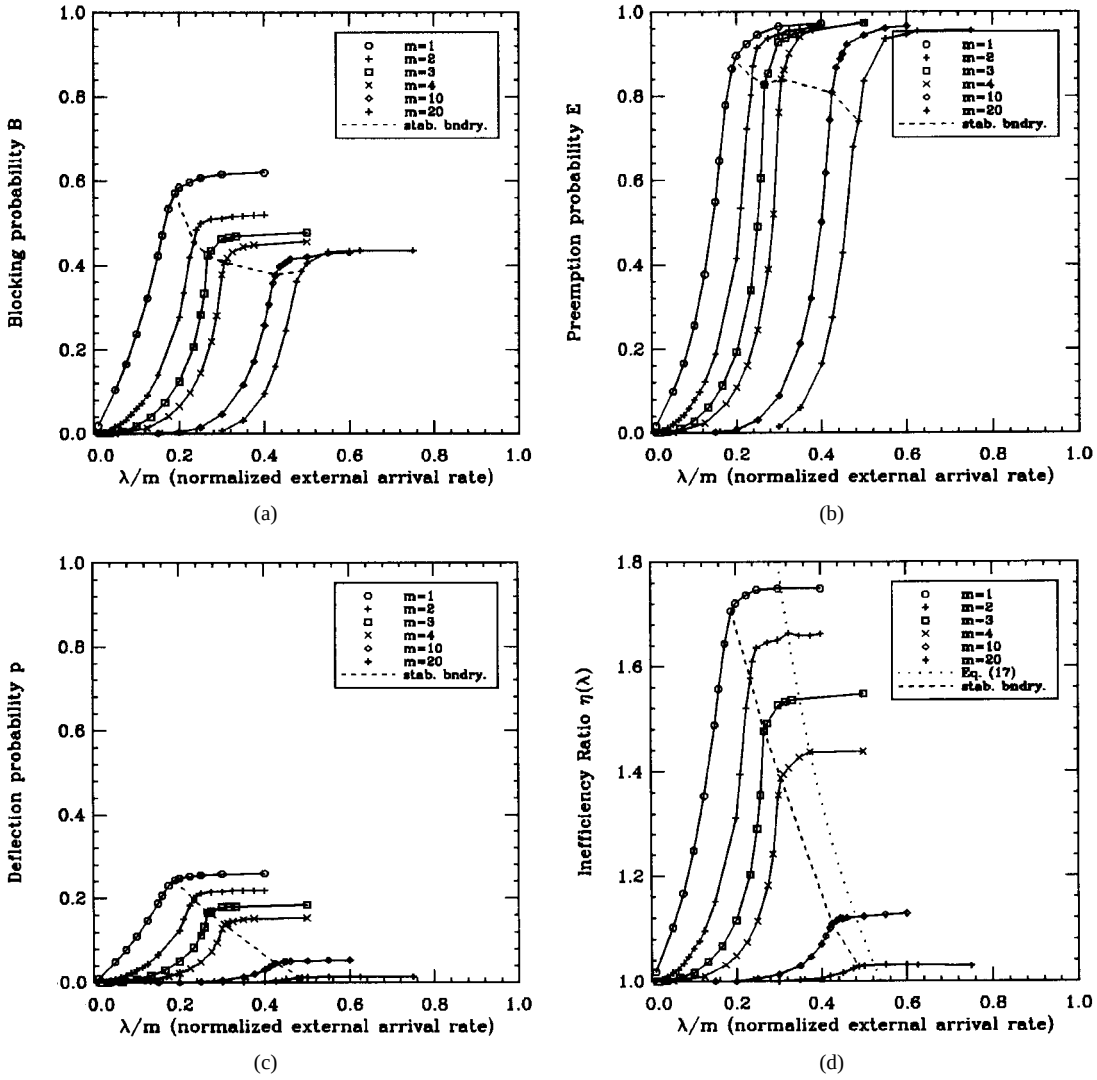
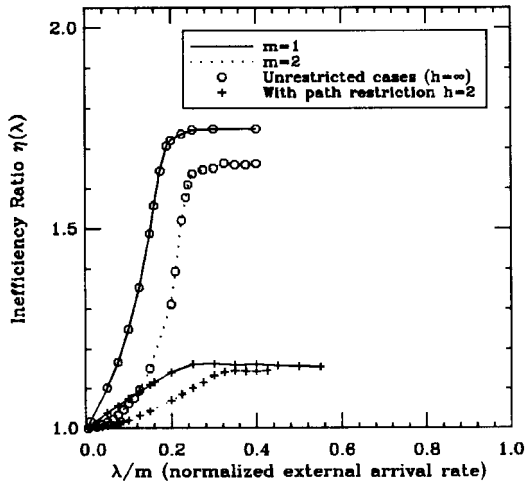


Fig. 10. We illustrate simulation results for the blocking probability B, the preemption probability E, the deflection probability p, and the inefficiency ratio $\eta(\lambda)$, as a function of λ/m for an 8 × 8 MS network, and several values of m. The dashed lines correspond to the stability boundary; points to the left of the boundary correspond to stable operation, and points to the right of the boundary correspond to unstable operation. The second (upper) dashed line in (d) corresponds to the necessary condition on stability given by (17).

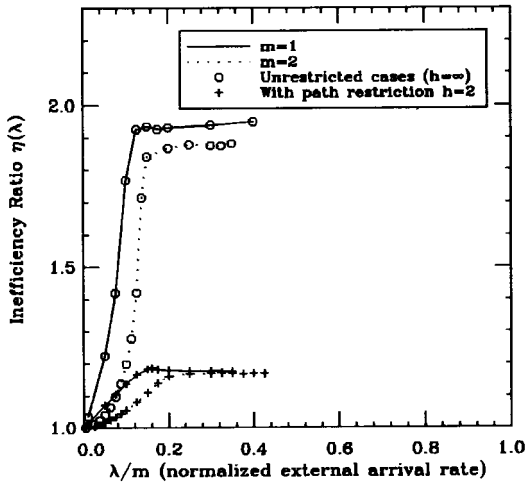
use capacity should be similar. The analogy between the two cases, however, is lost when $m \neq 1$. The preemption and the splitting of sessions, which are necessary for the VCD protocol to work, play no role in the unslotted deflection scheme, where deflections happen on a per packet basis, and each packet can be fully buffered at a node.

The results presented in Figs. 9 and 10 assume that there is no constraint on the lengths of the paths taken by the sessions. To reduce the waste of resources that arises when sessions follow very long paths, it is reasonable to impose an upper bound on the path lengths that are allowed. Fig. 11 illustrates the simulation results obtained for the case where the length of the path followed by a session is restricted to be at most h times the shortest distance between the source and the destination of the session (here, sessions that violate this condition are dropped and scheduled to retry at a later time). Note that for a given load λ, the improvement in efficiency obtained by using such a rule is significant, especially when m is small.

In certain deflection schemes the throughput does not increase monotonically with the rate at which nodes attempt to establish a connection, but it starts decreasing (slightly) after some point. To see whether this is also the case for the VCD protocol, we have plotted in Fig. 12 the normalized throughput; that is, the average number of sessions per node per unit of time that terminate normally. Note that the throughput increases with attempt rate until it reaches a plateau, so that no additional flow-control mechanism is necessary (at least not for performance reasons). This is because in the VCD protocol, preemptions actually have a positive influence on network efficiency by keeping the probability of deflections low. Using arguments similar to those used to determine the stability condition, an upper bound on the maximum normalized throughput is given by $\lambda_{\max}/m = 2\mu/D(0)$. For small values of m, the normalized throughput is not particularly satisfactory (for $m = 1$, the normalized throughput is only 35% of the upper bound), but it increases rapidly as m increases. The linear increase in link capacity m corresponds



(a)



(b)

Fig. 11. We illustrate the inefficiency factor $\eta(\lambda)$ for a 6×6 [Fig. 11(a)] and an 8×8 [Fig. 11(b)] MS network with link capacities $m = 1$ and 2, for the case $h = 2$, and compare it with the inefficiency factor of the corresponding unrestricted cases $(h = \infty)$.

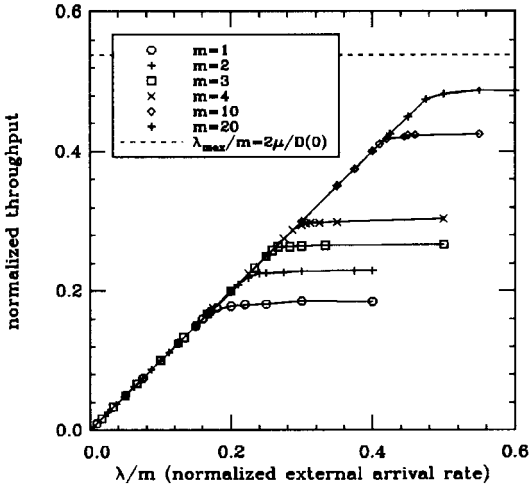


Fig. 12. We illustrate the normalized throughput as a function of λ/m for a 6×6 MS network and several values of m. We also illustrate the upper bound $(2\mu)/(D(0))$ on the throughput.

to a better-than-linear increase in the throughput (for $m = 10$, the maximum throughput is nearly 80% of the upper bound), indicating that the VCD protocol becomes more efficient as m increases.

VII. Conclusion

The virtual circuit deflection protocol presented in this paper compares favorably to wait-for-reservation and backpressure-based protocols, since it can provide lossless communication with little buffering at the switches and little pretransmission delay. The VCD protocol is a hybrid of virtual circuit switching and deflection routing, combining some of their individual advantages. The VCD protocol alleviates to a large extent the resequencing problem associated with datagram deflection schemes. Also, its small buffer requirements make it particularly appropriate for multigigabit networks that use optical switching, as the simple design given in Section IV indicates. We have presented analytical and simulation results on the throughput, the average path length, and other performance parameters of interest for the VCD protocol in an MS network. We believe that the results obtained are indicative of the performance of the protocol for other topologies of interest (provided that they offer a large number of alternative paths for a given source–destination pair), and they indicate that the VCD protocol is a viable connection and flow-control protocol for multigigabit and general data networks.

References

[1] B. Awerbuch, I. Cidon, I. Gopal, M. Kaplan, and S. Kutten, "Distributed control for PARIS," in *Proc. 9th Annu. Assoc. Comput. Mach. Symp. Principles of Distributed Computing*, Aug. 1990, pp. 145–159.

[2] I. Cidon and I. Gopal, "PARIS: An approach to integrated high-speed private networks," *Int. J. Digital Analog Cabled Syst.*, vol. 1, no. 2, pp. 77–86, Apr.–June 1988.

[3] N. F. Maxemchuk, "Problems arising from deflection routing: Live-lock, lock-out, congestion and message reassembly," *High-Capacity Local and Metropolitan Area Networks*, G. Pujolle, Ed. Berlin, Germany: Springer-Verlag, June 1990.

[4] F. Borgonovo, L. Fratta, and J. Bannister, "Unslotted deflection routing in all-optical networks," in *Proc. IEEE GLOBECOM'93*, vol. 1, pp. 119–125.

[5] Z. Haas and D. R. Cheriton, "Blazenet: A packet-switched wide-area network with photonic data path," *IEEE Trans. Commun.*, vol. 38, pp. 818–829, June 1990.

[6] F. Borgonovo and L. Fratta, "Deflection networks: Architectures for metropolitan and wide area networks," *Computer Networks and ISDN Syst.*, vol. 24, no. 2, pp. 171–183, Apr. 1992.

[7] I. Chlamtac and A. Fumagalli, "An optical switch architecture for Manhattan networks," *IEEE J. Select. Areas Commun.*, vol. 11, pp. 550–559, May 1993.

[8] F. Forghieri, A. Bononi, and P. R. Prucnal, "Analysis and comparison of hot-potato and single-buffer deflection routing in very high bit rate optical mesh networks," *IEEE Trans. Commun.*, vol. 43, pp. 88–98, Jan. 1995.

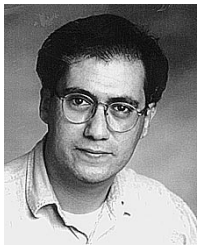
[9] R. L. Cruz and J.-T. Tsai, "COD: Alternative architectures for high speed packet switching," *IEEE/ACM Trans. Networking*, vol. 4, pp. 11–21, Feb. 1996.

[10] E. A. Varvarigos, "The 'Packing' and the 'Scheduling' packet switch architectures for almost all-optical lossless networks," *J. Lightwave Technol.*, vol. 16, pp. 1757–1767, Oct. 1998.

[11] A. G. Greenberg and J. Goodman, "Sharp approximate models of adaptive routing in mesh networks," in *Proc. Teletraffic Analysis and Computer Performance Evaluation*, June 1986, pp. 255–270.

[12] A. G. Greenberg and B. Hajek, "Deflection routing in hypercube networks," *IEEE Trans. Commun.*, vol. 40, pp. 1070–1081, June 1992.

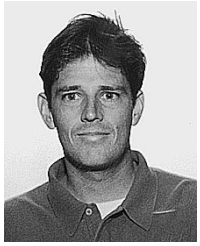
[13] J. Bannister, F. Borgonovo, L. Fratta, and M. Gerla, "A versatile model for predicting the performance of deflection-routing networks," *Perform. Eval.*, vol. 16, nos. 1–3, pp. 201–222, Nov. 1992.
[14] F. Borgonovo, L. Fratta, and J. Bannister, "On the design of optical deflection-routing networks," in *Proc. IEEE INFOCOM'94*, vol. 1, pp. 120–129.
[15] N. F. Maxemchuk, "Comparison of deflection and store-and-forward techniques in the Manhattan Street and Shuffle-Exchange networks," in *Proc. IEEE INFOCOM'89*, vol. 3, pp. 800–809.
[16] J. T. Brassil, "Deflection routing in certain regular networks," Ph.D. dissertation, Univ. California at San Diego, 1991.
[17] A. Krishna and B. Hajek, "Performance of shuffle-like switching networks with deflection," in *Proc. IEEE INFOCOM'90*, vol. 2, pp. 473–480.
[18] E. A. Varvarigos and D. P. Bertsekas, "Performance of hypercube routing schemes with or without buffering," *IEEE/ACM Trans. Networking*, vol. 2, pp. 299–311, June 1994.
[19] E. A. Varvarigos and V. Sharma, "An efficient reservation connection control protocol for gigabit networks," *Computer Networks and ISDN Syst.*, vol. 30, no. 12, pp. 1135–1156, July 1998.
[20] D. Bertsekas and R. Gallager, *Data Networks.* Englewood Cliffs, NJ: Prentice-Hall, 1992.
[21] A. Butner and D. Skirmont, "Architecture and design of a 40 gigabit per second ATM switch," in *Proc. ICCD'95*, pp. 352–357.
[22] E. A. Varvarigos and V. Sharma, "The ready-to-go virtual circuit protocol: A loss-free protocol for multigigabit networks using FIFO buffers," *IEEE/ACM Trans. Networking*, vol. 5, pp. 705–718, Oct. 1997.
[23] Z. Haas and R. D. Gitlin, "Field coding: A highspeed 'almost-all' optical interconnect," in *Proc. 25th Annu Conf. Inform. Sci. Syst.*, Mar. 1991.
[24] Z. Haas, "The 'Staggering Switch': An electronically controlled optical packet switch," *J. Lightwave Technol.*, vol. 11, pp. 925–936, May/June 1993.
[25] P. O'Reilly, "The case for circuit switching in future wide bandwidth networks," in *Proc. IEEE ICC'88*, vol. 2, pp. 899–904.
[26] P. J. Lin and R. G. Gallager, "Time slot interchange using fiber delay lines," 1995, preprint.
[27] A. Krishna, "Communication with few buffers: Analysis and design," Ph.D. dissertation, Univ. of Illinois at Urbana-Champaign, Dec. 1990.
[28] S. J. Golestani, "Congestion-free communication in high-speed packet networks," *IEEE Trans. Commun.*, vol. 39, pp. 1802–1812, Dec. 1991
[29] A. E. Eckberg, D. T. Luan, and D. M. Lucantoni, "An approach to controlling congestion in ATM networks," *Int. J. Dig. Analog Commun. Syst.*, vol. 3, no. 3, pp. 199–209, Apr.–Jun. 1990.



Emmanouel A. Varvarigos (M'92) received the Dipl.Eng. from the National Technical University of Athens, Athens, Greece, in 1988, the M.S. degree in electrical engineering in 1991, and the Ph.D. degree in electrical engineering and computer science in 1992, both from the Massachusetts Institute of Technology, Cambridge, MA.

In 1990, he was with Bell Communications Research, Morristown, NJ. In 1992, he joined the Department of Electrical and Computer Engineering, University of California at Santa Barbara, where he is currently an Associate Professor. In 1998–1999, during his sabbatical leave, he was a Visiting Associate Professor at the Technical University of Delft, The Netherlands, and the University of Patras, Greece. His research interests are in the areas of algorithms and protocols for data networks, performance evaluation, all-optical networks, parallel and distributed computation, and mobile communications.

Dr. Varvarigos received the First Panhellenic Prize in the Greek Mathematic Olympiad in 1982, the Technical Chamber of Greece Award four times (1984–1988), and the National Science Foundation Research Initiation Award in 1993. He was a co-organizer of the 1996 Workshop on Advanced Communication Systems and Networking.



Jonathan P. Lang (S'89) received the B.S. degree in electrical engineering from the University of California at San Diego in 1993, and the M.S. degree in electrical and computer engineering from the University of California at Santa Barbara in 1995, where he is completing the Ph.D. degree in electrical and computer engineering.

During the spring and summer of 1995, he was with Whitetree, Inc., Palo Alto, CA, where he developed rate-allocation algorithms for ATM switches. His research interests are in all-optical networking, protocol design, and optical switch architectures.

Dr. Lang is a member of Tau Beta Phi and Eta Kappa Nu.