ELSEVIER

Computer Networks 32 (2000) 211–227

COMPUTER NETWORKS

www.elsevier.com/locate/comnet

A reservation protocol for broadcast WDM networks and stability analysis ☆

Vijay Sivaraman [a], George N. Rouskas [b,*]

[a] *Department of Computer Science, University of California at Los Angeles, Los Angeles, CA 90095, USA*
[b] *Department of Computer Science, College of Engineering, North Carolina State University, P.O. Box 7534, Raleigh, NC 27695-7534, USA*

Abstract

We consider the problem of coordinating access to the various channels of a single-hop wavelength division multiplexing (WDM) network. We present a high performance reservation (HiPeR-ℓ) protocol specifically designed to overcome the potential inefficiencies of operating in environments with non-negligible processing, tuning, and propagation delays. HiPeR-ℓ differs from previous reservation protocols in that each control packet makes reservations for all data packets waiting in a node's queues, thus significantly reducing control overhead. Packets are scheduled for transmission using algorithms that can effectively mask the tuning times. HiPeR-ℓ also uses pipelining to mask processing times and propagation delays; parameter ℓ of the protocol is used to control the degree of pipelining. We use Markov chain theory to obtain a sufficient condition for the stability of the protocol. The stability condition provides insight into the factors affecting the operation of the protocol, such as the degree of load balancing across the various channels, and the quality of the scheduling algorithms. The analysis is fairly general, as it holds for MMBP-like arrival processes with any number of states, and for non-uniform destinations. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Single-hop optical networks; Wavelength division multiplexing (WDM); Reservation protocols; Markov modulated Bernoulli process (MMBP)

1. Introduction

Wavelength division multiplexing (WDM) is the most promising technology for bridging the gap between the speed of electronics and the virtually unlimited bandwidth available within the optical medium [8,10]. The single-hop WDM network architecture [14] is especially appealing because of the fact that, once information is transmitted as light in such a network, it will remain in the optical form until it reaches the destination. In a single-hop network, both a transmitter at the source and a receiver at the destination must operate on the same wavelength for a successful packet transmission. Thus, the problem of coordinating access to the various wavelengths of the network arises. This problem is further complicated by the fact that, in ATM-like local area networks (characterized by very high data rates and very small packet sizes), propagation delays, processing times, and transceiver

☆ An earlier version of this work was presented at the IEEE INFOCOM '97 Conference, Kobe, Japan, 1997.

* Corresponding author. Tel.: +1-919-515-3860, fax: +1-919-515-7925.

E-mail address: rouskas@csc.ncsu.edu (G.N. Rouskas).

1389-1286/00/$ - see front matter © 2000 Elsevier Science B.V. All rights reserved.
PII: S1389-1286(99)00137-1

tuning times all become non-negligible, and may actually be significantly larger than the packet transmission time.

In this paper, we present HiPeR-ℓ, a new reservation protocol for coordinating access to the various channels of a single-hop WDM local area network. HiPeR-ℓ will be used as the media access control (MAC) protocol for the DARPA-sponsored Helios regional optical network testbed that is currently being developed jointly by NCSU, MCNC and Lucent. The protocol is specifically designed to overcome the potential inefficiencies of operating in environments with non-zero processing, tuning, and propagation delays. The novelty of HiPeR-ℓ lies in the fact that, by transmitting a single control packet, nodes can make reservations for multiple data packets. Thus, control overhead is significantly reduced, and nodes can use scheduling algorithms that can effectively mask tuning times [19]. HiPeR-ℓ also uses pipelining to mask processing times and propagation delays; parameter ℓ (the *look-ahead*) of the protocol controls the degree of pipelining. Drawing upon results from Markov chain theory, we obtain a sufficient condition for the stability of the protocol that provides insight into the factors affecting the protocol's operation. In the analysis, we assume arrival processes that capture the notion of burstiness and the correlation of interarrival times, two important characteristics of traffic in high speed networks [18].

In the next section, we review some of the multiple access protocols for single-hop WDM networks. In Section 3 we present the network and traffic model. In Section 4 we describe HiPeR-ℓ, and in Section 5 we carry out a stability analysis. In Section 6 we present some numerical results, and we conclude the paper in Section 7.

2. Why a new media access control protocol?

Access to the various channels of a single-hop network is usually based on reservation schemes that require the use of control channels [5,6,9,11,12,23]. Existing protocols require that control information be transmitted on the control channel for *each* packet sent on the data channels. In *tell-and-go* protocols [5,11] the data packet is sent on the node's home channel immediately after the transmission of the corresponding control information. Thus, receiver collisions may arise and explicit acknowledgments are needed. Other protocols are *tell-and-wait* in nature [9,12,22]; nodes send the control information and wait for the control slot to reach all receivers. Then, they process the information in the control slot to determine if a data slot has been reserved for them. In the event of a successful reservation, the packet is transmitted in the corresponding slot and channel. In effect, the control slot information in tell-and-wait schemes is used by the individual nodes to build a picture of the packet queues at all other nodes in the network. Decisions about which packets to be transmitted next are taken in a distributed fashion based on protocol-specific rules common to all nodes.

The above protocols suffer from two problems:

- The control channel represents an *electronic processing bottleneck* [11] as control information for N packets must be received and processed for *each packet transmission and reception*. At the envisioned Gigabit per second data rates this processing overhead can be significantly greater than the packet transmission time for anything but networks of trivial size.
- All protocols operate by scheduling a *single* packet from each transmitter at a time (typically, the head-of-line packet). This packet is scheduled *independently* of other packets waiting for transmission at the same node. Hence, one transmitter and/or receiver tuning time is incurred for each packet transmission/reception.

The processing and tuning overhead associated with *each* packet on the data channels severely affects the throughput and delay performance of the network. To get a feeling of the magnitude of this problem, consider a 1 Gigabit per second ATM LAN. In such a system, a 1 μs transmitter tuning latency corresponds to more than two times the ATM cell transmission time. Further suppose that the time needed to process a control slot is equal to one cell transmission time. Then, an overhead of three cell times is incurred for each cell transmitted, bringing the maximum achievable throughput down to 25%!

A protocol that overcomes the processing bottleneck by introducing $k > 1$ control channels was presented in [11]. Its main drawback, however, is lack of scalability, as it requires $N + k$ wavelengths. In fact, almost all control channel protocols require a number of wavelengths at least equal to the number of nodes N. The MaTPi protocol [23] uses pipelining to mask the effect of tuning times. The PROTON protocol [12] can operate with any number of wavelengths, and its design explicitly considers tuning and processing times. However, PROTON schedules one packet at a time, and the results in [12] confirm the intuition that high processing and tuning times have a significant effect on delay and throughput.

The distributed queue multiple wavelength (DQMW) protocol [13] can also operate with any number of wavelengths, and considers tuning times when scheduling packets. DQMW attempts to overcome the head-of-line blocking of other protocols by considering multiple packets for transmission by a given node. But these packets are scheduled independently of each other, thus a tuning overhead is incurred for each. DQMW also has higher processing requirements than other protocols, since two control packets must be sent for each data packet. FatMAC [21] is a reservation protocol that does not require a separate control channel. Instead, all channels operate in cycles, with each cycle consisting of a control and data phase. Reservations are transmitted in the control phase, and the corresponding data packets are sent in the following data phase. Reservations are made only for the head-of-line packets, thus a control and tuning overhead is incurred for each data packet.

In this paper we present HiPeR-ℓ, a reservation protocol that has the following important features:

- It is scalable, as it can operate with any number of channels $C \leqslant N$.
- It may operate without a control channel, thus all channels are available for data transmission and no extra hardware is needed to monitor and access a control channel; this feature is especially useful when only a limited number of wavelengths can be supported. Control packets are transmitted *in-band* over the same channels used for data.
- It requires tunability only at one end, and is symmetric, in the sense that it can be easily implemented using either tunable transmitters or tunable receivers (in contrast, other protocols either require tunability at both ends [11–13] , or are asymmetric, i.e., they can operate only when tunability is provided at a particular end [5]).
- It ensures that packet transmissions are free of channel and receiver collisions.
- It schedules multiple packets for transmission by a node on a given channel using the algorithms in [19] which mask the tuning latency. Also, its control requirements are very low, since a single control packet can be used to make reservations for a multiple data packets.
- It uses pipelining to (a) overlap processing (i.e., the computation of a schedule) with packet transmissions, and (b) hide the effects of propagation delay. Furthermore, the degree of pipelining can be explicitly controlled through parameter ℓ of the protocol.

3. System and traffic model

3.1. Network model

We consider an optical broadcast WDM network with N nodes, each employing one transmitter and one receiver, as shown in Fig. 1. There are C wavelengths in the network, $\lambda_1, \ldots, \lambda_C$, with $C \leqslant N$. There is no separate control channel; all channels are used for data transmission, as well as for communicating control information. Without loss of generality, we only consider tunable-transmitter, fixed-receiver networks. Each tunable transmitter can tune to, and transmit on any wavelength. The fixed receiver at station j, on the other hand, is assigned a home channel $\lambda(j) \in \{\lambda_1, \ldots, \lambda_C\}$.

The network is packet-switched, with fixed-size packets. The buffer space at each node is partitioned into C independent queues. Each queue contains packets destined for receivers which listen to a particular wavelength. This arrangement eliminates the head-of-line blocking problem, and permits a node to send a number of packets back-to-back when tuned to a particular channel. The

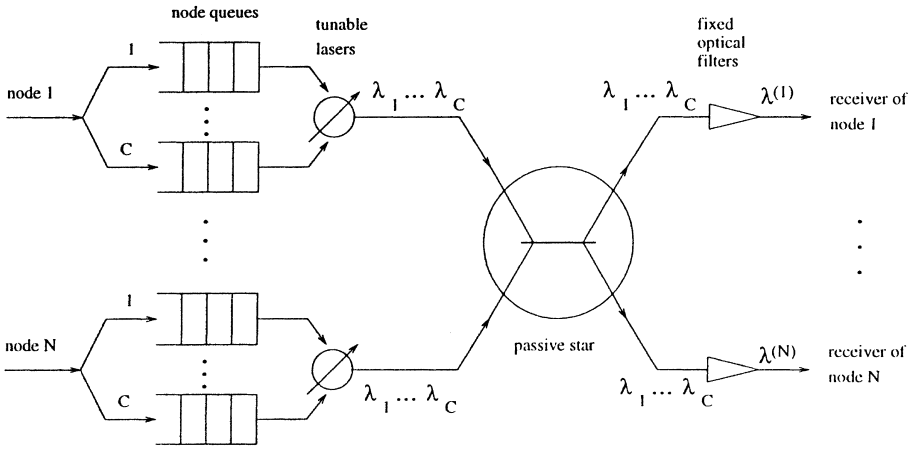


Fig. 1. Network architecture with N nodes and C channels.

network operates in a slotted mode, with a slot time equal to a packet transmission time. All nodes are synchronized at slot boundaries. Packets buffered at the cth queue of each node are transmitted on a FIFO basis into the optical medium on wavelength λ_c.

We let integer $\Delta \geqslant 1$ denote the number of slots a tunable transmitter takes to tune from one wavelength to another. We also let τ denote the one way propagation delay between a pair of nodes.

3.2. Transmission schedules

One of the potentially difficult issues that arise in a WDM environment is that of packet scheduling in the presence of non-negligible tuning latencies [1,4,15,17,19]. In [19] we showed that careful scheduling can mask the effects of arbitrarily long tuning latencies. The key idea is to have each tunable transmitter send a *block* of packets on each wavelength before switching to the next. Doing so makes it possible to overlap the tuning latency at a node with packet transmissions from other nodes. The main result of [19] was a set of new algorithms for constructing near-optimal (and, under certain conditions, optimal) schedules for transmitting a set of traffic demands $\{a_{ic}\}$. Quantity a_{ic} represents the number of packets to be transmitted by node i onto channel λ_c. The schedules are such that no collisions ever occur. They are also easy to implement in a high speed environment, since the order in which the various nodes transmit is the same for all channels [19].

Fig. 2 illustrates the part of such a schedule corresponding to channel λ_c. Each node i is assigned a_{ic} *contiguous* slots for transmitting packets on that channel. These a_{ic} slots are followed by a *gap* of $g_{ic} \geqslant 0$ slots during which no node may transmit on λ_c. This gap may be necessary to ensure that node $i+1$ has sufficient time to tune from wavelength λ_{c-1} before starting transmission on λ_c. However, the algorithms in [19] are such that the number of slots in most of the gaps is equal to either zero or a small integer. Thus, the length of the schedule is very close to the lower bound. The scheduling algorithms in [19] require complete information about the traffic demands $\{a_{ic}\}$. HiPeR-ℓ is a reservation protocol used by the network nodes to dynamically share this information.

3.3. Traffic model

The performance analysis of protocols for WDM networks has been typically carried out assuming uniform traffic and Poisson arrivals. However, to study correctly the performance of the network, one needs to use traffic models that capture the notion of burstiness and correlation, two important characteristics of traffic in high speed networks, and which permit non-uniform destinations [18]. To this end, we assume that the

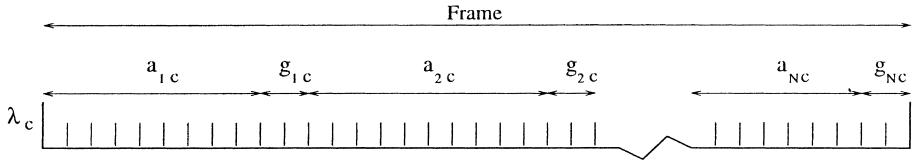


Fig. 2. Part of the schedule corresponding to packet transmissions on channel λ_c.

arrival process to each node is characterized by a two-state Markov modulated Bernoulli process (MMBP), hereafter referred to as 2-MMBP. This is a Bernoulli process whose arrival rate varies according to a two-state Markov chain. (For details on the properties of the 2-MMBP, the reader is referred to [16].) We note that all of our results can be readily extended to MMBPs with more than two states. The 2-MMBP for node $i, i = 1, \ldots, N$, is characterized by the transition matrix $\boldsymbol{Q}_i$, and by $\boldsymbol{A}_i$ as follows:

$$\boldsymbol{Q}_i = \begin{bmatrix} q_i^{(00)} & q_i^{(01)} \\ q_i^{(10)} & q_i^{(11)} \end{bmatrix}; \quad \boldsymbol{A}_i = \begin{bmatrix} \alpha_i^{(0)} & 0 \\ 0 & \alpha_i^{(1)} \end{bmatrix}. \tag{1}$$

In (1), $q_i^{(kl)},\ k, l = 0, 1$, is the probability that the 2-MMBP will make a transition to state l, given that it is currently at state k. Obviously, $q_i^{(k0)} + q_i^{(k1)} = 1,\ k = 0, 1$. Also, $\alpha_i^{(0)}$ and $\alpha_i^{(1)}$ are the arrival rates of the Bernoulli process at states 0 and 1, respectively. The arrival process to each node i is given by a different 2-MMBP, independent of the arrival processes to other nodes. From [16] we obtain the average arrival rate γ_i of the ith 2-MMBP as

$$\gamma_i = \frac{q_i^{(10)} a_i^{(0)} + q_i^{(01)} a_i^{(1)}}{q_i^{(01)} + q_i^{(10)}}. \tag{2}$$

γ_i is the probability that any slot contains a packet, regardless of the state of the 2-MMBP. We only consider 2-MMBPs for which

$$q_i^{(kl)} > 0, \quad k, l = 0, 1,\ \ i = 1, \ldots, N. \tag{3}$$

Conditions (3) guarantee that the two-state Markov chain of each 2-MMBP is irreducible and aperiodic, thus it has a stationary distribution.

We let r_{ij} denote the probability that a new packet arriving to node i will have j as its destination node. We will refer to $\{r_{ij}\}$ as the *routing probabilities*. This description implies that the routing probabilities are source node dependent and non-uniformly distributed.

4. Description of the HiPeR-ℓ protocol

The operation of HiPeR-ℓ is rather simple:

- Each network node periodically sends control packets informing all other nodes about its traffic demands.
- Each node has a copy of the packet scheduling algorithm developed in [19]. Upon receipt of all control packets transmitted by other nodes, each node independently runs the algorithm to determine at what time slots to transmit its own data packets.

There are two main differences between HiPeR-ℓ and any of the protocols that have appeared in the literature. First, in HiPeR-ℓ a node does not send a reservation request for its head-of-line packet only. Instead, each control packet of a node i contains information about *all* the packets that were queued in any of i's C queues at a certain instant in time. By sending a control packet, node i is in effect making reservations for all packets it had waiting for transmission at that instant. The next time node i is scheduled to transmit on wavelength λ_c, it will send a number of data packets back-to-back equal to the number of reservations it made for this channel in the corresponding previous control packet. Secondly, control packets are not transmitted over a separate channel. Reservations are *in-band* over the same channels used for data. Furthermore, time in the channels is not divided into distinct reservation and data phases as in FatMAC [21]. Exactly when control packets are transmitted will be discussed shortly.

The next subsection describes a first version of HiPeR-ℓ. We then extend the protocol by introducing pipelining to mask the effects of long propagation delays and processing times.

4.1. The basic idea: HiPeR-1

The basic operation of HiPeR-ℓ is illustrated in Fig. 3. For reasons that will become apparent shortly, we will refer to this version of the protocol as HiPeR-1.

Assume that, somehow, each node i has made reservations for $a_{ic}^{(k)}$ data packets on wavelength λ_c, and that these reservations are known to all nodes. Each node independently runs the scheduling algorithm in [19] to compute a packet transmission schedule. However, the input to this algorithm is not quantities $\{a_{ic}^{(k)}\}$, but rather quantities $\{a_{ic}^{(k)}+1\}$; the extra slot is for transmitting a control packet (more on this shortly). The algorithm will allocate $a_{ic}^{(k)}+1$ *contiguous* slots to node i for transmission to destinations listening on wavelength λ_c. We will call this allocation of slots to source–wavelength pairs a *frame*.

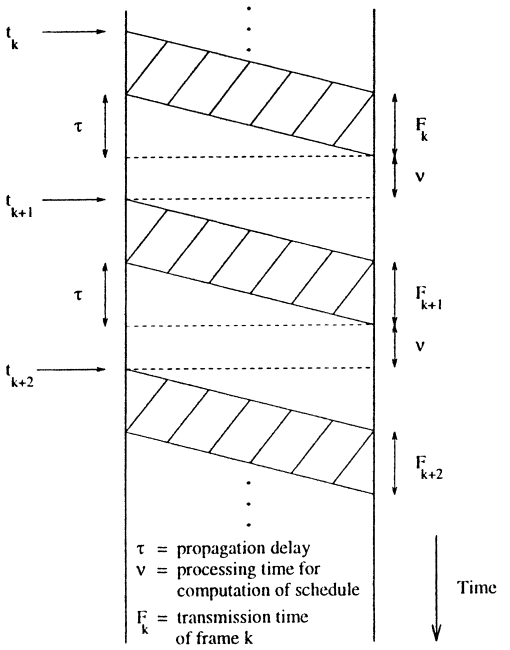


Fig. 3. Operation of HiPeR-ℓ when the look-ahead $\ell = 1$.

Suppose now that at time t_k in Fig. 3 all nodes have constructed the kth frame from the known quantities $\{a_{ic}^{(k)}+1\}$. Transmission of this frame can then begin at time t_k. Consider the $a_{ic}^{(k)}+1$ slots in the frame allocated to node i for transmissions on channel λ_c. Node i will transmit only $a_{ic}^{(k)}$ data packets in these slots (this is the number of data slots it had reserved). In the last slot node i will transmit a control packet with information about the number of data packets that were in its C queues at the beginning of the frame (i.e., at time t_k), excluding packets it transmits during this frame. In other words, a control packet from node i in frame k carries C integers, $a_{i1}^{(k+1)},\ldots,a_{iC}^{(k+1)}$, and is used to make reservations for future transmissions on each channel. An identical copy of the control packet is transmitted by node i on each wavelength, and carries a special address recognized by all receivers in the network. As a result, by the time the last packet of the frame reaches all receivers, each node has complete information (although somewhat dated) of the queue status at all nodes. Each node can then use this information to run the scheduling algorithm anew to determine the *next* frame.

Let F_k be the length, in slots, of the kth frame; F_k includes the Δ slots required for tuning the transmitters to their initial channels. Referring to Fig. 3 we note that at time $t_k+F_k+\tau$ all nodes will have access to the control information transmitted in frame k (recall that τ denotes the propagation delay). Let ν denote the time it takes to run the scheduling algorithm to construct the next frame. [1] At time $t_{k+1}=t_k+F_k+\tau+\nu$, the transmission of frame $k+1$ may start. At the same time, each node i will record the number of packets in each of its C queues, and will use that information for constructing its control packets for frame $k+1$. In effect, the value of a_{ic} in a control packet transmitted in frame $k+1$ represents the number of packets that arrived to the cth queue of node i between time t_k (the beginning of

[1] One important aspect of the scheduling algorithms in [19] is that their running time depends only on system parameters such as the number of nodes and channels, *not* on the actual frame length.

transmission of frame k) and time t_{k+1} (the beginning of transmission of frame $k+1$).

As described, the protocol is said to have a *look-ahead* $\ell = 1$, since control information transmitted during the kth frame is used to construct the $(k+1)$th frame; thus the name HiPeR-1. This protocol falls into the class of *gated* reservation schemes [3], since only those packets that arrived prior to the beginning of frame k will be transmitted in frame $k+1$. The difference between HiPeR-ℓ and traditional reservation protocols (including FatMAC [21]) is that HiPeR-ℓ does not have a distinct reservation phase. Instead, control packets are transmitted within a frame along with data packets. This is necessary in order to minimize the tuning overhead. If there was a separate reservation phase, the transmitters would have to (a) tune to each channel during the reservation phase to transmit a single control message, and (b) tune to each channel during the data phase to transmit the data packets.

In our discussion so far, we have assumed that the size of a control packet is equal to that of a data packet. This is a reasonable assumption for networks with small data packets. Let B be the size of each of the C queues at each node. Since a control packet carries the size of each queue, its length is equal to $C \log_2 B$ bits plus the header. If the size of each data packet is significantly larger than $C \log_2 B$ bits, it would be inefficient to use a data slot for transmitting the small amount of control information required. It is possible, however, to overcome this inefficiency as follows. Let L be an integer such that the size of each data packet is L times the size of the control packet, and assume that the unit of time (slot) in the network is the control packet transmission time. When a node i makes reservations for a_{ic} packets, it is allocated $La_{ic} + 1$ slots which are sufficient for transmitting a_{ic} data packets and one control packet. Without loss of generality, in the following we only consider the case where control and data packets have the same size.

4.2. Masking processing and propagation delays through pipelining

Observe in Fig. 3 that there are no transmissions in an interval of size $\tau + \nu$ between the end of frame k (at time $t_k + F_k$) and the beginning of frame $k+1$ (at time t_{k+1}). If quantity $\tau + \nu$ is small compared to the average transmission time of a frame, a system running HiPeR-1 will achieve a reasonable throughput. In a high data rate environment, however, processing and propagation delays may be significantly long. As a result, the basic protocol of Fig. 3 will experience long idle times with severe effects on overall throughput. We now show how pipelining can solve this problem and keep channel utilization at high levels.

Pipelining can be introduced in the protocol by using values of look-ahead greater than one. Fig. 4 illustrates the operation of HiPeR-ℓ when the look-ahead $\ell = 4$. Let us consider frame $k+1$ whose transmission starts at time t_{k+1}. Control packets transmitted within this frame carry information about the number a_{ic} of data packets that arrived to the various queues in the interval $[t_k, t_{k+1})$. However, this information is not used for constructing frame $k+2$. As we see in Fig. 4, the information carried by the control packets transmitted in frame $k+1$ has not been processed until

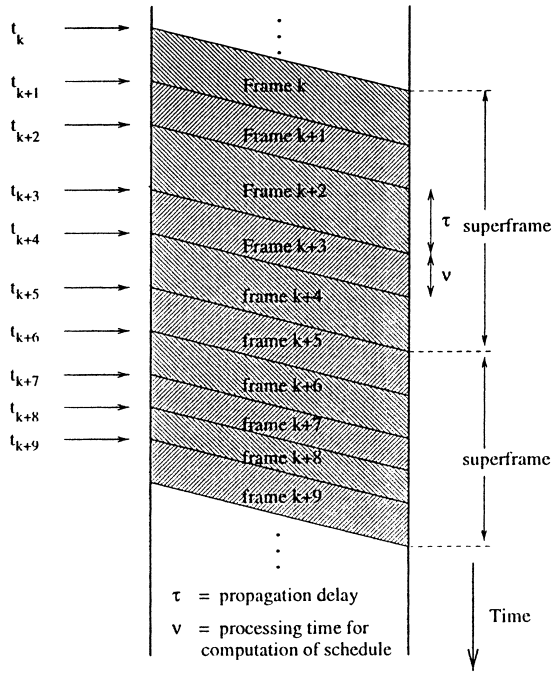


Fig. 4. Operation of HiPeR-ℓ when the look-ahead $\ell = 4$.

after time t_{k+4} when frame $k+4$ starts. Thus, this information is used to construct frame $k+5$ whose transmission starts at time t_{k+5}. In general, we have the following rule:

> When the look-ahead is $\ell \geqslant 1$, the control packets of each frame k carry information about the data packets that arrived during the previous frame $k-1$. This information is used to construct frame $k+\ell$.

As Fig. 4 indicates, by selecting an appropriate value for the look-ahead ℓ, we can ensure that a frame is ready for transmission immediately after the end of the previous frame, thus keeping channel utilization at high levels. Let $\bar{F}$ denote the *average* frame transmission time. Then, the value of the look-ahead should be selected as

$$\ell = \left\lceil \frac{\tau + v}{\bar{F}} \right\rceil. \tag{4}$$

Note, however, that (4) is *not* sufficient to guarantee that no idling will occur. Because of the stochastic nature of the system, it is possible that during a relatively long period of time, only a few packets arrive. If, as a result of such a behavior, the transmission time of a number of successive frames is smaller than the processing time v, then idling will occur. This is due to the fact that control information in a frame cannot be processed until after the schedule based on control packets in the previous frame has been completed. Thus, if a series of very short frames are transmitted, the processing times will dominate, causing some channel idling. There are two ways to overcome this problem. The first, suggested by the authors of PROTON [12], is to employ multiple processing resources at each node so that they can process control information of more than one frames in parallel. Alternatively, it is sufficient that the processing time v be smaller than the transmission time of the smallest possible frame, one carrying only control packets (N packets per channel). However, even if none of these approaches is possible, we do not expect channel idling to be a problem if the look-ahead ℓ is selected as (4) specifies. Unless the network operates at very low loads, the probability of having multiple consecutive short frames is very low, and thus, the propagation and processing times will be overlapped most of the time.

HiPeR-ℓ incurs an overhead of NC control packets for each frame transmitted (each node sends one control packet on each wavelength). In terms of efficiency, this overhead is not expected to be a problem except at very low data rates when a frame may carry a small number of data packets. The advantage of *in-band* reservations over control channel-based schemes is that all available wavelengths can be used to transmit data, and no extra hardware is needed to monitor and access the control channel. If necessary, however, HiPeR-ℓ can be easily adapted to use *out-of-band* reservation messages. In this case, for each frame of data packets a node needs to send exactly one control packet on the control channel. Thus, only a small fraction of the control channel capacity is needed for reservation messages; the remaining capacity can be used for other purposes, such as network management, synchronization, etc.

Finally, we note that HiPeR-ℓ, as many other reservation protocols for single-hop networks, relies on distributed computations to create and maintain a network-wide packet queue, based on which packet transmissions are scheduled. Since all nodes use the same algorithm and the same input values obtained from the control packets, they will all compute the same schedule. As long as all nodes are functioning correctly, this mode of operation will work quite well. However, even one node that breaks down and starts transmitting without respect to the common rules will create chaos. In these situations, techniques and mechanisms are needed to detect and isolate malfunctioning nodes, and to restore normal operation even in a subset of the available wavelengths. The problem of devising such techniques and mechanisms is beyond the scope of this paper, and should be explored in future research.

5. Performance analysis

An analysis of TDMA schemes in which a node is allocated multiple consecutive slots per frame has been carried out in [20]. There, the generating

functions of the queue size and of the delay distribution are derived for fairly general arrival processes. The model in [20] assumes a fixed TDMA frame size, with each node receiving a fixed number of slots occupying the same positions in every frame. In HiPeR-ℓ, however, each node will make reservations for a different number of slots from frame to frame. Consequently, the frame size will vary. Furthermore, the scheduling algorithm is run anew for each frame. Therefore, the order in which the various nodes transmit may be different in consecutive frames. As a result, the analysis in [20] is not applicable here.

For the same reasons, an exact delay analysis of a system running HiPeR-ℓ appears to be difficult. We note, however, that packet delay is directly related to the frame size. In the following, we carry out a stability analysis of HiPeR-ℓ and obtain a sufficient condition on the total arrival rate to the network for the frame size to remain bounded. Although in our analysis we assume that the arrival process to each node is described by a 2-MMBP, it can be easily seen that the same condition applies to other MMBP-like processes with a larger number of states.

Before we proceed, we note that there are two factors that directly affect the operation of a network running HiPeR-ℓ: the degree of load balancing across the various channels, and the quality of the scheduling algorithm used. In order to quantify their effect on the performance of the protocol, we define two parameters, as follows:

- *Degree of load balancing* $\epsilon_b \geqslant 0$. Let A_k be the total number of data packets arriving to the network nodes within frame k. Each of these packets will be transmitted on one of the C channels in a future frame. If the load is perfectly balanced across the C channels, each channel will carry exactly A_k/C of these packets. In general, the traffic load will not be perfectly balanced. Parameter ϵ_b is defined so as to provide an upper bound on the number of packets to be carried by any single channel. Specifically, for any frame k, no more than $(1+\epsilon_b)(A_k/C)$ of the packets arriving during that frame are destined for any given channel. Under perfect load balancing, $\epsilon_b = 0$. The degree of load balancing ϵ_b can be controlled if slowly tunable, rather than fixed receivers are used. Then, as the traffic pattern changes, dynamic balancing techniques may be employed, i.e., nodes may be assigned new receive wavelengths, so as to keep the load evenly spread across all channels [2].
- *Scheduling guarantee* $\epsilon_s \geqslant 0$. Let $\widehat{F}_k$ be the lower bound on the length of frame k, based on the data reservations made in a previous frame. Parameter ϵ_s is defined such that, the algorithm used to schedule packet transmissions will, *on the average*, [2] construct a frame of length at most $(1+\epsilon_s)\widehat{F}_k$.

5.1. Markov chain model

Consider a network running HiPeR-ℓ with a look-ahead $\ell \geqslant 1$, as shown in Fig. 4. We will call a collection of $\ell+1$ consecutive frames a *superframe*. Our analysis below is based on the observation (refer to Fig. 4) that the data packets transmitted within a superframe are exactly those packets that arrived to the various network nodes during the previous superframe.

We analyze the system by constructing its underlying Markov chain (MC) embedded at superframe boundaries. We observe the system at an instant just before the beginning of a new superframe. The state of the system is described by the tuple $(x, \underline{y})$, where

- x represents the length, in slots, of the superframe that is about to be transmitted $(x = 0, 1, 2, 3, \ldots)$.
- $\boldsymbol{y}$ is a vector $\boldsymbol{y} = (y_1, \ldots, y_N)$, with y_i indicating the state of the arrival process to node i $(y_i = 0, 1,\ i = 1, \ldots, N)$.

As the state of the system evolves in time, it defines a MC $\mathscr{M}$. To see this, let $(x, \boldsymbol{y})$ be the current state of the system, and $(x', \boldsymbol{y}')$ be the state at the beginning of the next superframe. Obviously, the new state $\boldsymbol{y}'$ of the arrival processes

[2] As we shall see in the proof of Lemma 5.2, only the *average-case* behavior of the scheduling algorithm needs to be known (and it can be determined empirically). This result is important since it is well-known [7] that most scheduling algorithms can be expected to do much better than their *worst-case* bounds.

depends only on the current state $\boldsymbol{y}$ and the number of slots x that will elapse. The length x' of the new superframe depends on (a) the number of arrivals during the current superframe and how these packets are distributed across the various channels, (b) the number of control packets to be transmitted within the superframe, and (c) the scheduling algorithm used. The number of arrivals in the current superframe depends only on the state $\boldsymbol{y}$ of the arrival processes at the beginning of the superframe, and its length x. The number of control packets transmitted within a superframe is $(\ell+1)CN$, since the superframe consists of $\ell+1$ individual frames. The scheduling algorithm used is independent of the system state. Therefore, the new length x' also depends only on the current state $(x,\boldsymbol{y})$.

Let $P[(x,\boldsymbol{y}) \to (x',\boldsymbol{y}')]$ denote the probability that the system makes a transition to state $(x',\boldsymbol{y}')$, given that it is currently in state $(x,\boldsymbol{y})$. (Given the description of the N 2-MMBPs, the value ℓ of the look-ahead, and the scheduling algorithm, the transition probabilities are completely specified. However, as we shall shortly see, the exact values of these transition probabilities are not necessary in our analysis.) It is now straightforward to verify that, if conditions (3) hold, MC $\mathscr{M}$ is irreducible and aperiodic. Thus, $\mathscr{M}$ will have a stationary distribution if we can find scalars $\{\pi_{(x,\boldsymbol{y})}\}$ such that $\sum_{(x,\boldsymbol{y})} \pi_{(x,\boldsymbol{y})} = 1$, and they satisfy

$$\pi_{(x,\boldsymbol{y})} = \sum_{(x',\boldsymbol{y}')} \pi_{(x',\boldsymbol{y}')} P[(x',\boldsymbol{y}') \to (x,\boldsymbol{y})], \quad \forall (x,\boldsymbol{y}). \tag{5}$$

Solving Eqs. (5) by inspection requires writing out the actual values of the transition probabilities, a complicated task. However, we are only interested in obtaining a condition for MC $\mathscr{M}$ to have a stationary distribution. We now observe that random variable $\underline{y}$ can take exactly $K = 2^N$ values which we will denote by $\boldsymbol{y}_1,\ldots,\boldsymbol{y}_K$. Let us partition the state space of MC $\mathscr{M}$ into subsets S_x of states with the same superframe length: $S_x = \{(x,\boldsymbol{y}_1),\ldots,(x,\boldsymbol{y}_K)\}$. We construct a new MC $\mathscr{M}'$ embedded at superframe boundaries, with state space $\{S_x\}$, and transition probabilities $P_{x,x'}$, where $P_{x,x'}$ is equal to the transition probability in MC $\mathscr{M}$ from the states in S_x to the states in $S_{x'}$,

$$P_{x,x'} = \sum_{x,\boldsymbol{y} \in S_x} \pi_{(x,\boldsymbol{y})} \sum_{x',\boldsymbol{y}' \in S_{x'}} P[(x,\boldsymbol{y}) \to (x',\boldsymbol{y}')]. \tag{6}$$

We now prove the following lemma.

Lemma 5.1. *MC $\mathscr{M}$ has a stationary distribution iff MC $\mathscr{M}'$ also has a stationary distribution.*

Proof. In the forward direction, suppose that there exist positive scalars $\{\pi_{(x,\boldsymbol{y})}\}$ that satisfy (5) and sum up to 1. It is straightforward to see that MC $\mathscr{M}'$ will have a stationary distribution $\{\pi_x\}$, where $\pi_x = \sum_{\boldsymbol{y}} \pi_{(x,\boldsymbol{y})}$, $\forall x$. In the reverse direction, suppose that $\mathscr{M}'$ has a stationary distribution $\{\pi_x\}$. Since each time state S_x of MC $\mathscr{M}'$ is entered the random variable $\boldsymbol{y}$ will have one of K possible values, there will exist positive scalars $\{\pi_{(x,\boldsymbol{y}_i)}\}$, $i = 1,\ldots,K$, whose sum will equal π_x, and which will satisfy (5). □

We are now ready to prove our main result.

Lemma 5.2. *Let $\gamma = \sum_{i=1}^{N} \gamma_i$ be the total arrival rate to the network, and suppose that we have*

$$\gamma < \frac{C}{(1+\epsilon_b)(1+\epsilon_s)}. \tag{7}$$

Then, MC $\mathscr{M}'$ has a stationary distribution.

Proof. Let D_x denote the drift at state x of $\mathscr{M}'$. Because of Pake's lemma [3, 3A.5], in order to show that $\mathscr{M}'$ has a stationary distribution, we only need to show that there exist a state $x_0 \geqslant 0$ and a scalar $\delta > 0$ such that,

$$D_x \leqslant -\delta, \quad \forall x > x_0. \tag{8}$$

The drift at state x of MC $\mathscr{M}'$ can be written as

$$D_x = E[x'|x] - x, \tag{9}$$

where $E[x'|x]$ is the expected length of the next superframe given that the length of the current superframe is x slots.

The expected number of packets that arrive in the current superframe of size x slots, independently of the state of the arrival processes at the beginning and end of the superframe, is γx, where γ is the sum of the arrival rates to the network nodes. Because of the definition of parameter ϵ_b,

no more than $(1+\epsilon_b)(\gamma x/C)$ of these arriving packets are destined for any given channel. In addition, there are $(\ell+1)N$ control packets that will be transmitted on each wavelength within the next superframe. Therefore, the expected number of packets (data plus control) transmitted on any channel during the next superframe cannot be greater than $(1+\epsilon_b)(\gamma x/C)+(\ell+1)N$. Because of the definition of parameter ϵ_s, the length of this next superframe cannot be greater than $(1+\epsilon_s)$ times this last quantity. Therefore, we can bound the expected length of the next superframe by

$$E[x'|x] \leqslant (1+\epsilon_s)\frac{(\ell+1)NC+(1+\epsilon_b)\gamma x}{C}. \tag{10}$$

If we use this expression in (9), we obtain an upper bound on the drift at state x,

$$D_x \leqslant (1+\epsilon_s)\frac{(\ell+1)NC+(1+\epsilon_b)\gamma x}{C} - x. \tag{11}$$

After some algebraic manipulation of (11), we find that (8) is satisfied if we let

$$x_0 = \left\lceil \frac{\delta+(1+\epsilon_s)(\ell+1)N}{1-(1+\epsilon_b)(1+\epsilon_s)(\gamma/C)} \right\rceil. \tag{12}$$

This x_0 is positive iff (7) holds. □

Finally, by combining Lemmata 5.1 and 5.2 we obtain the desired result,

Corollary 5.1. *If the total arrival rate satisfies* (7) *then MC* $\mathscr{M}$ *has a stationary distribution.*

The stability condition (7) is simple yet powerful, as it provides insight into the two main factors that determine the performance of the network, namely, the degree of load balancing, and the quality of the scheduling algorithm. As we can see, the lower the degree of load balancing (i.e., the larger the value of ϵ_b in (7)), the lower the maximum arrival rate that the network can sustain (recall that C is the capacity of the network). Similarly with the scheduling efficiency, captured by parameter ϵ_s in (7). Although (7) was derived specifically for HiPeR-ℓ, we believe that these two factors play a similar role in any reservation protocol for single-hop networks.

Let $\bar{F}$ denote the mean frame size when the stability condition (7) is satisfied. From the definition of the look-ahead ℓ, a packet arriving during a frame k will be transmitted to its destination within frame $k+\ell+1$. We can then obtain the following expression for the mean packet delay $\bar{D}$:

$$\bar{D} = (\ell+1)\bar{F}. \tag{13}$$

6. Numerical results

We demonstrate the operation of the HiPeR-ℓ protocol by considering two networks, each with $N=40$ nodes and $C=10$ channels. The two networks have the same broadcast architecture shown in Fig. 1, but they differ in how traffic is distributed across the destination nodes. The first network, hereafter referred to as the *uniform-routing* network, is such that the destination of a packet is uniformly distributed across all destinations,

$$r_{ij} = \frac{1}{39} \quad \forall i \neq j \quad \text{(uniform-routing network)} \tag{14}$$

The second network is a *client–server* network. There are two servers (nodes 1 and 2) and 38 clients (nodes 3 through 40). The routing probabilities are

$$r_{ij} = \begin{cases} 0 & i=j \\ 0.01 & i=1,\ j=2 \text{ or } i=2,\ j=1 \\ 0.99/38 & i=1,2,\ j=3,\ldots,40 \\ 0.114 & i=3,\ldots,40,\ j=1,2 \\ 0.772/38 & i,j=3,\ldots,40 \end{cases}$$
(client–server network) (15)

The arrival process to each of the nodes of either network is described by a different 2-MMBP. Since it is not practical to provide the matrices **Q** and **A** for the 40 2-MMBPs of each network, we instead show two important parameters for each 2-MMBP. In Fig. 5 we show the arrival rate $\gamma_i,\ i=1,\ldots,40$, in (2) of the 2-MMBPs describing the arrival process to each of the 40 nodes of the two networks. The total arrival rate to each network is $\gamma = 7.344$. In Fig. 6 we show the squared

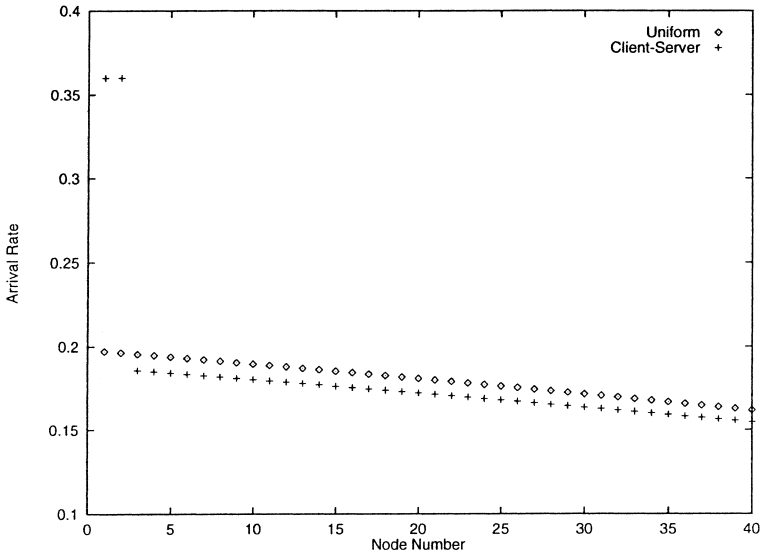


Fig. 5. Arrival rate of the arrival processes.

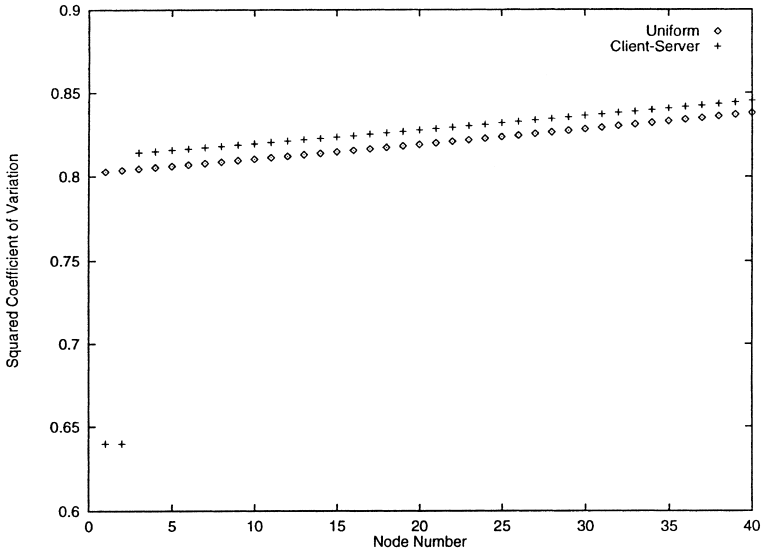


Fig. 6. Squared coefficient of variation of the interarrival time for the arrival processes.

coefficient of variation of the interarrival time obtained in [16]. As we can see, the arrival processes were selected so that the two parameters take a wide range of values.

Based on the results of the previous section, we have assigned receive wavelengths to the various nodes so as to spread the traffic evenly across the channels. For the uniform-routing network this can be achieved by simply assigning each of the 10 wavelengths to exactly four receivers. In the client–server network, however, there is more traffic entering the two servers. Therefore, we have decided to assign one wavelength to each of the two servers, while the remaining eight wavelengths are

shared by the other 38 nodes (six of these wavelengths are each shared by five nodes, while the other two are each shared by four nodes).

We have run a number of simulations to determine the frame size and mean packet delay in these networks running HiPeR-ℓ for various values of the look-ahead ℓ. In our simulations we assume that the propagation delay $\tau = 20$ slots, the processing time $v = 100$ slots, and the tuning latency $\Delta = 4$ slots. Figs. 7 and 8 plot the actual and mean frame size of the uniform-routing and client–server network, respectively, when the look-ahead $\ell = 1$.

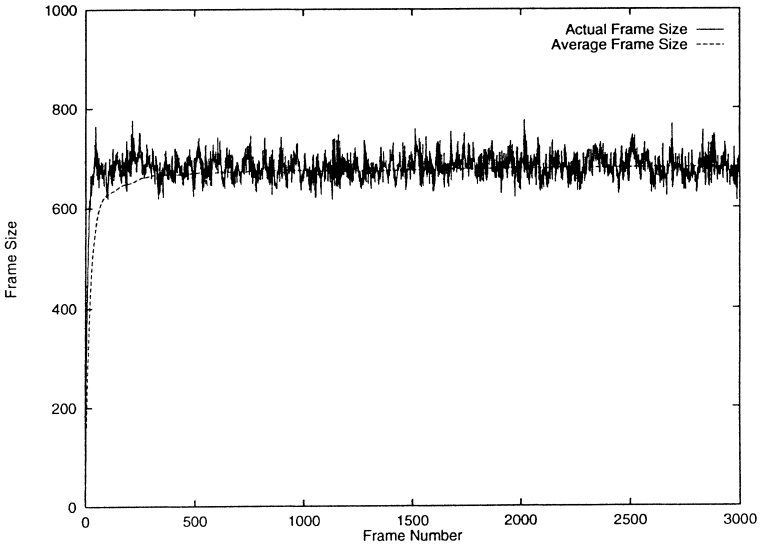


Fig. 7. Frame size of the uniform-routing network when the look-ahead is $\ell = 1$.

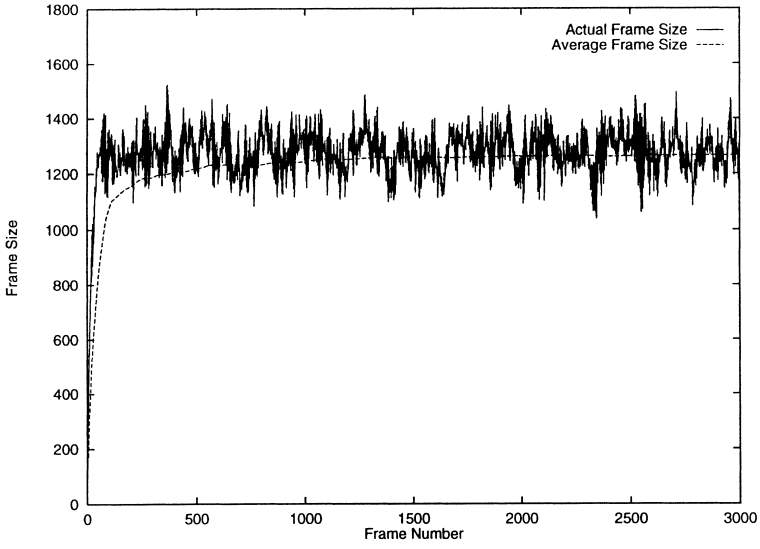


Fig. 8. Frame size of the client–server network when the look-ahead is $\ell = 1$.

The size of the first 3000 frames in the simulation is plotted. We can see that the mean is well-defined, and that the size of individual frames oscillates around this mean, as expected. Also, both the average and the actual frame size is larger for the client–server network, due to the high traffic concentrated on the two channels allocated to the server nodes.

In Figs. 9 and 10 we plot the mean frame size as a function of throughput for the uniform-routing and the client–server network, respectively. The figures also show how the average frame size is affected by the value of the look-head, ℓ. For both networks, when ℓ is increased from 1 to 2, there is a significant decrease in the frame size. This can be explained by noting that, when the look-ahead is 1, there is an idle period after the end of each frame equal to $\tau + \nu = 120$ slots (refer also to Fig. 3). During this period, packets may arrive to the network nodes, but no packets are transmitted. Thus, the average frame size $\bar{F}$ has to be large enough so that, on average, the number of packets transmitted during $\bar{F}$ slots equals the number of packets arriving during $\bar{F} + 120$ slots. When $\ell = 2$, the propagation delay and processing time of 120 slots are completely overlapped with the transmission of the next frame, no idling occurs, and the frame size is smaller. Since $\ell = 2$ is sufficient to completely mask the 120 slots of propagation delay and processing time, there is nothing to gain from making $\ell = 3$ or 4, and the average frame size is not affected significantly.

In Figs. 11 and 12 we show the delay versus throughput curves for the uniform-routing and client–server network, respectively. The mean delay values are plotted with 95% confidence intervals, which, however, are so narrow that they are not visible. A look-ahead of 1 has the worst performance, as expected. Also, at low loads, a look-ahead $\ell = 3$ provides for shorter delays than a look-ahead $\ell = 2$, while the opposite is true for higher loads. At low loads, few packets arrive during a frame, thus the average frame size when $\ell = 2$ is not large enough to completely overlap the propagation delay and processing time. Thus, idling occurs after the end of two frames, and the result is longer delays than a look-ahead $\ell = 3$. As the load increases, the average frame size for $\ell = 2$ also increases. When the load is such that the 120 slots are completely masked with $\ell = 2$, no further gain is possible by using $\ell = 3$. That is, a look-ahead $\ell = 3$ will not decrease the frame size, but will increase the

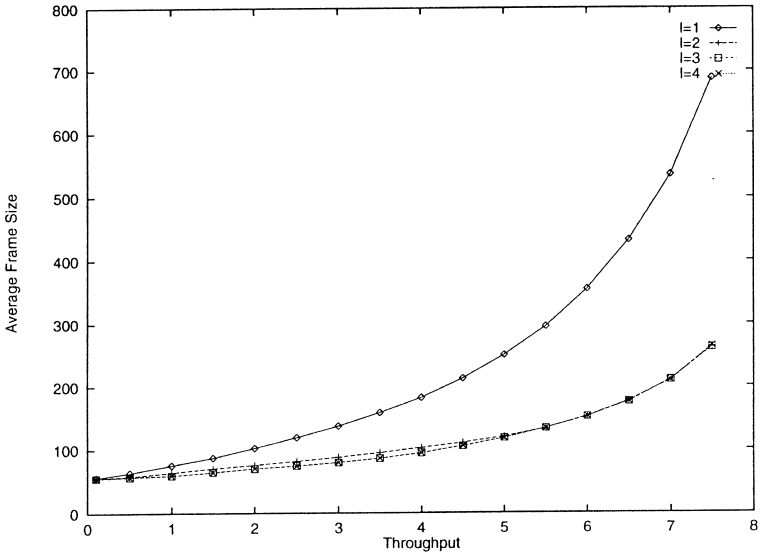


Fig. 9. Average frame size vs. throughput for the uniform-routing network.

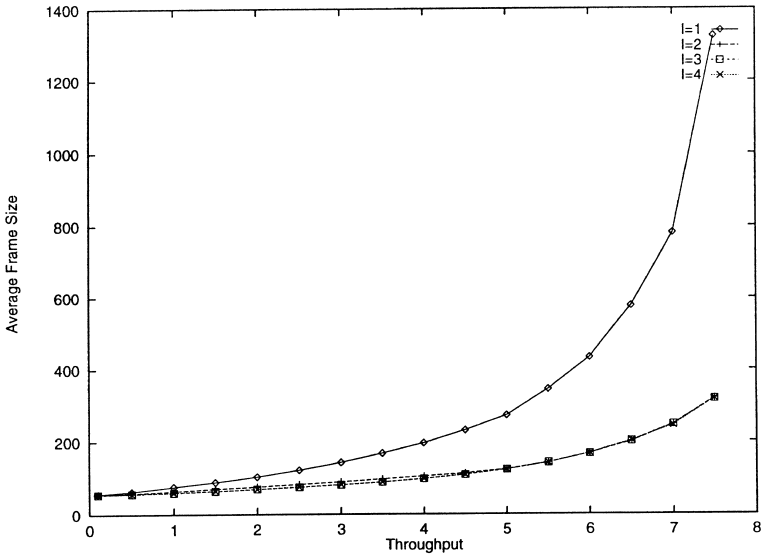


Fig. 10. Average frame size vs. throughput for the client–server network.

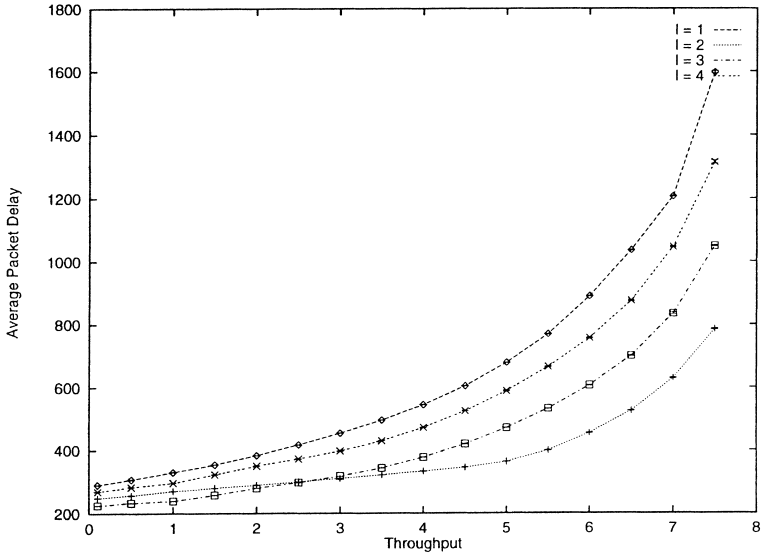


Fig. 11. Delay vs. throughput for the uniform-routing network.

delay, as seen from (13). Finally, a look-ahead of $\ell = 4$ or more offers no advantage compared to a look-ahead of $\ell = 3$, resulting in a higher delay.

Our results indicate that, in order to achieve the best performance possible, the value of the look-ahead must be carefully selected to ensure that propagation and processing times are completely overlapped. In our experiments, we have also observed that the frame size and the mean packet delay are mainly determined by the degree of load balancing and the quality of scheduling, in agreement with (7).

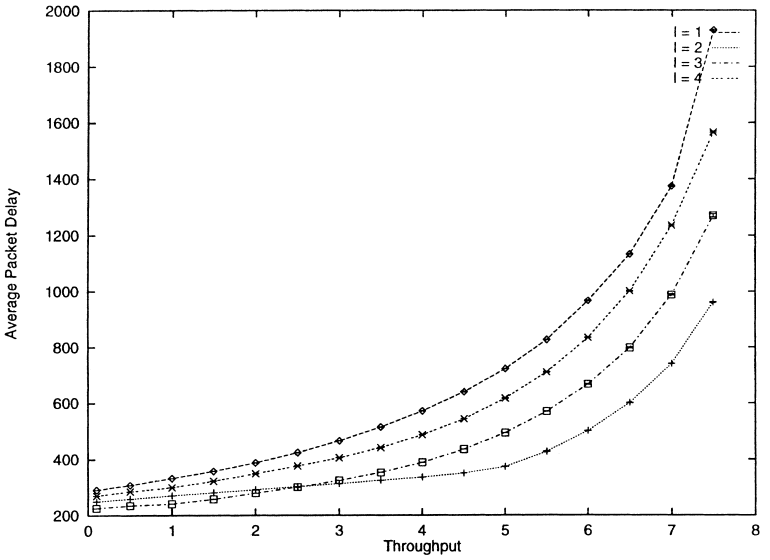


Fig. 12. Delay vs. throughput for the client–server network.

7. Concluding remarks

We have considered the media access problem arising in single-hop WDM networks. We introduced HiPeR-ℓ, a new reservation protocol designed to overcome the problems posed by non-negligible processing, tuning, and propagation delays. In HiPeR-ℓ, nodes send multiple reservation requests in a single control packet. As a result, the control requirements of the protocol are low, and nodes can use algorithms that schedule multiple packet transmissions on each wavelength, effectively masking the tuning latency. The parameter ℓ controls the degree of pipelining in the operation of the protocol, and can be used to mask the propagation delay and the processing time. We have derived a condition for the protocol to reach stability that mathematically captures the effect of load balancing and of the efficiency of the scheduling algorithm on the the overall network performance.

References

[1] M. Azizoglu, R.A. Barry, A. Mokhtar, Impact of tuning delay on the performance of bandwidth-limited optical broadcast networks with uniform traffic, IEEE J. Select. Areas Commun. 14 (5) (1996) 935–944.

[2] I. Baldine, G.N. Rouskas, Dynamic load balancing in broadcast WDM networks with tuning latencies, in: Proceedings of INFOCOM '98, IEEE, New York, March 1998, pp. 78–85.

[3] D. Bertsekas, R. Gallager, Data Networks, Prentice-Hall, Englewood Cliffs, NJ, 1992.

[4] M.S. Borella, B. Mukherjee, Efficient scheduling of non-uniform packet traffic in a WDM/TDM local lightwave network with arbitrary transceiver tuning latencies, IEEE J. Select. Areas Commun. 14 (5) (1996) 923–934.

[5] M.-S. Chen, N.R. Dono, R. Ramaswami, A media-access protocol for packet-switched wavelength division multiaccess metropolitan area networks, IEEE J. Select. Areas Commun. 8 (6) (1990) 1048–1057.

[6] R. Chipalkatti, Z. Zhang, A.S. Acampora, Protocols for optical star-coupler network using WDM: performance and complexity study, IEEE J. Select. Areas Commun. 11 (4) (1993) 579–589.

[7] E. Coffman, M.R. Garey, D.S. Johnson, An application of bin-packing to multiprocessor scheduling, SIAM J. Computing 7 (1) (1978) 1–17.

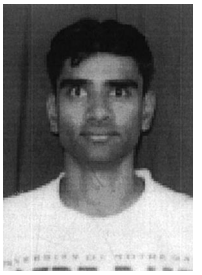
[8] R. Cruz, G. Hill, A. Kellner, R. Ramaswami, G. Sasaki, Y. Yamabayashi (Eds.), Optical Networks (special issue), IEEE J. Select. Areas Commun. 14 (5) (1996).

[9] E.M. Foo, T.G. Robertazzi, A distributed global queue transmission strategy for a WDM optical fiber network, in: Proceedings of INFOCOM '95, IEEE, New York, April 1995, pp. 154–161.

[10] M. Fujiwara, M. Goodman, M. O'Mahony, O. Tonguz, A. Willner (Eds.), Wavelength Technologies and Networks (special issue), J. Lightwave Technol. 14 (6) (1996).

[11] P.A. Humblet, R. Ramaswami, K.N. Sivarajan, An efficient communication protocol for high-speed packet-

switched multichannel networks, IEEE J. Select. Areas Commun. 11 (4) (1993) 568–578.
[12] D.A. Levine, I.F. Akyildiz, PROTON: a media access control protocol for optical networks with star topology, IEEE/ACM Trans. Networking 3 (2) (1995) 158–168.
[13] A. Muir, J.J. Garcia-Luna-Aceves, Distributed queue packet scheduling algorithms for WDM-based networks, in: Proceedings of INFOCOM '96, IEEE, New York, March 1996, pp. 938–945.
[14] B. Mukherjee, WDM-based local lightwave networks Part I: single-hop systems, IEEE Network Magazine (May 1992) 12–27.
[15] Z. Ortiz, G.N. Rouskas, H.G. Perros, Scheduling of multicast traffic in tunable-receiver WDM networks with non-negligible tuning latencies, in: Proceedings of SIGCOMM '97, ACM, New York, September 1997, pp. 301–310.
[16] D. Park, H.G. Perros, H. Yamashita, Approximate analysis of discrete-time tandem queueing networks with bursty and correleated input traffic and customer loss, Operations Res. Lett. 15 (1994) 95–104.
[17] G.R. Pieris, G.H. Sasaki, Scheduling transmissions in WDM broadcast-and-select networks, IEEE/ACM Trans. Networking 2 (2) (1994) 105–110.
[18] G. Pujolle, H.G. Perros, Queueing systems for modelling ATM networks, in: International Conference on the Performance of Distributed Systems and Integrated Comm. Networks, Kyoto, Japan, September 1991, pp. 10–12.
[19] G.N. Rouskas, V. Sivaraman, Packet scheduling in broadcast WDM networks with arbitrary transceiver tuning latencies, IEEE/ACM Trans. Networking 5 (3) (1997) 359–370.
[20] I. Rubin, Z. Zhang, Message delay analysis for TDMA schemes using contiguous-slot assignments, in: Proceedings of ICC '88, 1988, pp. 418–422.
[21] K. Sivalingam, P. Dowd, A multi-level WDM access protocol for an opticall interconnected multi-processor system, IEEE/OSA J. Lightwave Technol. 13 (11) (1995) 2152–2167.
[22] K. Sivalingam, J. Wang, Media access protocols for WDM networks with on-line scheduling, IEEE/OSA J. Lightwave Technol. 14 (6) (1996) 1278–1286.
[23] S. Tridandapani, J.S. Meditch, A.K. Somani, The MaTPi protocol: masking tuning times through pipelining in WDM optical networks, in: Proceedings of INFOCOM '94, IEEE, New York, June 1994, pp. 1528–1535.



Vijay Sivaraman received his B. Tech in Computer Science and Engineering from the Indian Institute of Technology at Delhi, India, in 1994, and his M.S. in Computer Science from North Carolina State University in 1996. He is currently pursuing his Ph.D. at UCLA. His research interests include packet scheduling issues and QoS support in high-speed networks.

George N. Rouskas received the Diploma in Electrical Engineering from the National Technical University of Athens (NTUA), Athens, Greece, in 1989, and the M.S. and Ph.D. degrees in Computer Science from the College of Computing, Georgia Institute of Technology, Atlanta, GA, in 1991 and 1994, respectively. He joined the Department of Computer Science, North Carolina State University in August 1994, and he has been an Associate Professor since July 1999. His research interests include high-speed and lightwave network architectures, multipoint-to-multipoint communication, and performance evaluation. He is a recipient of a 1997 NSF Faculty Early Career Development (CAREER) Award, and a co-author of a paper that received the Best Paper Award at the 1998 SPIE conference on All-Optical Networking. He also received the 1995 Outstanding New Teacher Award from the Department of Computer Science, North Carolina State University, and the 1994 Graduate Research Assistant Award from the College of Computing, Georgia Tech. He is a co-guest editor for the IEEE Journal on Selected Areas in Communications, Special Issue on Protocols and Architectures for Next Generation Optical WDM Networks, and is on the editorial board of the Optical Networks Magazine. He is a member of the IEEE, the ACM and of the Technical Chamber of Greece.