A Simulation Study of Access Protocols for Optical Burst-Switched Ring Networks

Lisong Xu, Harry G. Perros, and George N. Rouskas

Department of Computer Science, North Carolina State University, Raleigh, NC, USA
lxu2,hp,rouskas@csc.ncsu.edu

Abstract. In this paper, we consider a WDM metro ring architecture with optical burst switching. Several access protocols are proposed and their performance is analyzed by simulation.

1 Introduction

Optical burst switching (OBS) [1,2,3,4,5,6] is a switching technique that occupies the middle of the spectrum between the well-known circuit switching and packet switching paradigms, borrowing ideas from both to deliver a completely new functionality. The unit of transmission is a *burst*, which may consist of several packets. The transmission of each burst is preceded by the transmission of a control packet, which usually takes place on a separate signaling channel. Unlike circuit switching, a source node does not wait for confirmation that a path with available resources has been set up; instead, it starts transmitting the data burst soon after the transmission of the control packet. We will refer to the interval of time between the transmission by the source node of the first bit of the control packet and the transmission of the first bit of the data burst as the *offset*. The control packet carries information about the burst, including the offset value, the length of the burst, its priority, etc. Based on this information, intermediate nodes configure their switch fabric to switch the burst to the appropriate output port. However, in case of congestion or output port conflicts, an intermediate node may drop a burst. Also, consecutive bursts between a given source-destination pair may be routed independently of each other.

There are several variants of burst switching, mainly differing on the length of the offset. The most well-known scheme is *Just Enough Time* (JET) [3], in which the offset is selected in a manner that takes into account the processing delays of the control packet at the intermediate switches. Let $T_i^{(p)}$ denote the processing delay of a control packet at an intermediate switch, $T_d^{(p)}$ denote the processing delay of a control packet at the destination switch, and $T_d^{(s)}$ denote the time to setup (configure) the destination switch. Then, the offset value for JET is :

$$\text{offset}_{\text{JET}} = \left(\sum_i T_i^{(p)}\right) + T_d^{(p)} + T_d^{(s)} \tag{1}$$

One issue that arises in computing the offset under JET is determining the number of intermediate switching nodes (hops) between the source and destination. Information about the number of hops in a path may not, in general, be readily available; even if it is known, it may not be valid when used. Thus, it is desirable to use an offset value that does not depend on the path used and does not require the exchange of information among network nodes.

The part of the offset value that depends on the path between the source and destination is the sum of the processing times at intermediate nodes. Given recent advances in hardware implementation of communication protocols, we can assume that the processing time $T_i^{(p)}$ in (1) will be very short for most common functions of the signaling protocol. In this case, fiber delay lines may be used at intermediate nodes to delay each incoming burst by an amount of time equal to $T_i^{(p)}$. Then, the first term in the right hand side of (1) can be omitted when computing the offset. We call this new scheme the *Only Destination Delay (ODD)* protocol, and its offset is given by:

$$\text{offset}_{\text{ODD}} \quad = \quad T_d^{(p)} \; + \; T_d^{(s)} \tag{2}$$

Instead of using destination-specific values for the processing and switching delays in (2), one may use a constant offset value by taking the maximum of these values over all destinations. Such a value may significantly simplify the design and implementation of signaling protocols and optical switches for burst switching networks [2].

In this paper we study burst switching protocols for ring networks. Our focus on ring topologies is motivated by the wide deployment of optical rings. These networks represent a significant investment on the part of carriers, and are currently being upgraded to support WDM. Section 2 describes the ring network we consider and the basic operation of burst switching in such an environment. Section 3 provides a detailed description of the various burst switching access protocols studied in this paper. Section 4 presents the simulation results on the performance of these burst switching access protocols, and finally Section 5 provides some concluding remarks.

2 The Ring Network under Study

We consider N OBS nodes organized in a unidirectional ring, as shown in Figure 1. Each fiber link supports $N+1$ wavelengths. Of these, N wavelengths are used to transmit bursts, and the $(N+1)$-th wavelength is used as the control channel. Each OBS node is attached to one or more access networks. In the direction from the access networks to the ring, the OBS node acts as a concentrator. Buffered packets are grouped together and transmitted in a burst to the destination OBS node. A burst can be of any size between a minimum and maximum value. Bursts travel along the ring without undergoing any electro-optic conversion at intermediate nodes. In the other direction, from the ring to the access networks, an OBS node terminates optical bursts, electronically processes the data packets contained therein, and delivers them to users.

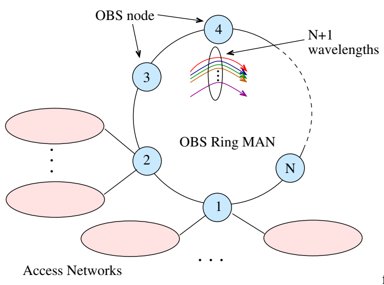

Fig. 1. OBS Ring MAN

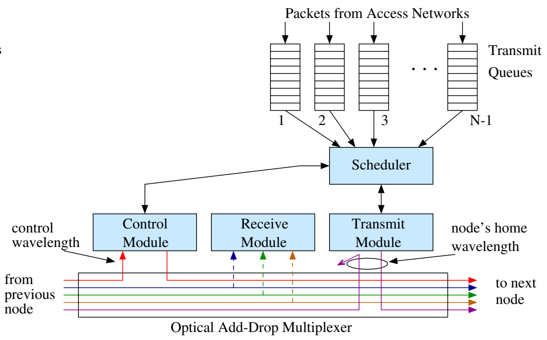

Fig. 2. Node architecture (delay lines not shown)

The architecture of an OBS node is shown in Figure 2. Each node is equipped with one optical add-drop multiplexer (OADM), and two pairs of optical transceivers. The first pair consists of a receiver and transmitter tuned to the control wavelength, and is part of the control module in Figure 2. The control wavelength is dropped by the OADM at each node, and added back after the control module has read the control information and has inserted new information.

The second pair of transceivers consists of a transmitter that is fixed tuned to the node's *home wavelength*, and an agile receiver that can receive from all N data wavelengths. Each OBS node has a dedicated home wavelength on which it transmits its bursts. The OADM at each node removes the optical signal from the node's home wavelength by dropping the corresponding wavelength, as Figure 2 illustrates. The OADM also drops the optical signal on other burst wavelengths, whenever they contain bursts for this node. In the case where multiple bursts arrive, each on a different wavelength, at an OBS node, the receive module in Figure 2 employs a collision resolution strategy to determine which burst will be accepted. To support ODD, an extra fiber delay line (not shown in Figure 2) is added into the node to delay outgoing data on all wavelengths except the control wavelength and the node's home wavelength.

Packets waiting for transmission are organized into transmit queues according to their destination. The order in which transmit queues are served is determined by the scheduler module in Figure 2. We assume that the transmit queues are considered in a Round-Robin manner.

The control wavelength is used for the transmission of control slots. In a ring with N nodes, N control slots, one for each node, are grouped together in a *control frame* which continuously circulates around the ring. Depending on the length of the circumference of the ring, there may be several control frames circulating simultaneously. In this case, control frames are transmitted back-to-back on the control wavelength. Each node is the owner of one control slot in each control frame. Each control slot contains several fields, as Figure 3 illustrates. The format and type of the fields depend on the OBS protocol used. In general, however, each control slot includes fields for the destination address, the offset, and the burst size. Other fields may be included for some of the protocols.

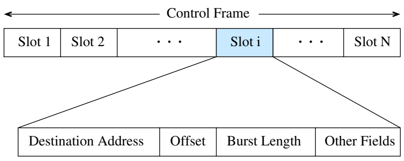

Fig. 3. Structure of a control frame

To transmit a burst, a node waits for the next control frame and writes the burst information (destination address, burst length, and offset) in its own control slot. If it has nothing to transmit, it just clears all the fields in its control slot. Each node also reads the entire control frame to determine whether any control slots indicate a burst transmission to this node. If so, and assuming that the node is not in the process of receiving another burst, it instructs its tunable receiver to tune to the appropriate wavelength to receive the burst; that is, preemption is not allowed. In case of a receiver collision (i.e., when the address of this node is specified in multiple control slots), the destination node selects one of the bursts to receive.

Each node acts as a source node (inserting bursts), as an intermediate node (passing through bursts to downstream nodes), and as a destination node (terminating bursts). As a result, each node must read each control frame in its entirety before determining what action to take. Therefore, in a ring network the time to process a control frame is the same for intermediate and destination nodes (i.e., $T_i^{(p)} = T_d^{(p)}$). The control frame is delayed by this amount of time as it passes through each node. This delay is the sum of the control frame transmission time plus the time to process the control frame, and it can be kept short by employing a simple protocol implemented in hardware. A number of OBS protocols having these features are described in the next section.

3 OBS Protocols

Since each OBS node is assigned a unique home wavelength, bursts may be lost due to receiver collisions. This occurs when two or more nodes transmit bursts to the same destination, and the burst transmissions overlap in time. We propose several access protocols which can be classified in three classes, depending on how receiver collisions are resolved.

1. *Source node.* The source node resolves receiver collisions using the information transmitted on the control wavelength.
2. *Destination node.* A source node must get permission from the destination to send a burst. Each destination schedules requests to avoid collisions.
3. *Token passing.* Tokens are used to resolve receiver collisions.

Our emphasis is on protocols that use few rules, are simple to implement in hardware and are distributed in nature. We have deliberately avoided protocols that are centralized in nature, or they require the collection of transmit queue sizes, or they require network-wide synchronization (e.g., TDM-based schemes). We assume that the maximum and minimum burst size that can be transmitted on the ring is specified by constants `MaxBurstSize` and `MinBurstSize`, respectively. Furthermore, a transmit queue is not *eligible* for service unless its size is at least equal to the value of `MinBurstSize`.

3.1 Round-Robin with Random Selection (RR/R)

The first protocol a round-robin scheduler at each node to serve the transmit queues, and lets each receiver randomly select a burst from the bursts that arrive simultaneously. We call this protocol *Round-Robin with Random Selection* (RR/R). The operation of the protocol at node i is as follows.

At the transmitting side, the scheduler of node i visits all eligible transmit queues, in a round-robin fashion. If at time t_1, transmit queue j is selected for service, then node i waits for the first control frame that arrives after time t_1. Then, node i writes the burst information and destination address j in its own control slot. After a delay equal to the offset value, node i transmits the burst on its home wavelength.

At the receiving side, when a control frame arrives at node i, it scans the control slots of the control frame, checking for any slot that has i in the destination address field. If more than one such slots are found, node i randomly selects one of them, say k. In this case, all bursts to node i except the one from node k will be lost. Node i then checks whether its receiver is free at the time when the burst from node k arrives at node i, and checks whether its receiver has enough time to tune to another wavelength. If so, it instructs its receiver to tune to node k's home wavelength in order to receive the burst transmission. Otherwise, it gives up on the burst from node k.

3.2 Round-Robin with Persistent Service (RR/P)

The *Round-Robin with Persistent Service* (RR/P) protocol is similar to the RR/R protocol, but it is designed to eliminate receiver conflicts that can be detected prior to the transmission of a burst. The operation of this protocol at node i is as follows.

At the transmitting side, node i maintains a variable `EarliestFreeTime(j)` for each destination node j, which specifies the earliest time at which the receiver of node j would be free. This variable is updated by monitoring the burst information in control slots that have j in the destination address field. The scheduler at node i visits all eligible transmit queues in a round-robin fashion. If at time t_1, transmit queue j is selected for service, then node i waits for the first control frame that arrives after time t_1. Suppose it arrives at time t_2, then node i updates the variable `EarliestFreeTime(j)` based on relevant information in the control frame. Node i also computes the time t_3 that the first bit of its burst would arrive at node j: $t_3 = t_2 + T_i^{(p)} + \text{offset} + \delta_{ij}$, where δ_{ij} is the burst propagation delay from node i to node j. If `EarliestFreeTime(j)` plus the receiver tuning time at node j is less than t_3, then node i writes its burst information in its own control slot, and sends the burst after a delay equal to the offset. If, on the other hand, `EarliestFreeTime(j)` plus the receiver tuning time at node j is greater than t_3, then sending the burst will result in a conflict. In this case, node i does not transmit the burst; instead it waits for the next control frame and repeats the process of transmitting the burst to node j. This is the *persistent* feature of the protocol, in that the round-robin scheduler does not proceed to serve the next transmit queue until the burst to node j has been sent. Note that deferring the transmission of a burst based on a calculation of the earliest free time for receiver j does not altogether eliminate receiver collisions.

At the receiving side, the operation of the protocol is identical to RR/R.

3.3 Round-Robin with Non-persistent Service (RR/NP)

The operation of the *Round-Robin with Non-Persistent Service* (RR/NP) protocol is identical to the operation of the RR/P protocol with one exception. Suppose that at time t_1 node i has selected transmit queue j for service using the RR scheduler. Suppose also that once the first control frame arrives after time t_1, the node determines that transmitting a burst to j would result in a collision. The node refrains from transmitting the burst, and proceeds to serve the next eligible transmit queue upon arrival of the next control frame.

The RR/NP protocol may result in lower delay than RR/P. However, since a node gives up its burst transmission whenever it determines that it will lead to a collision, RR/NP may lead to the starvation of certain transmit queues, and thus, it has fairness problems. Also, RR/NP does not completely eliminate receiver collisions.

3.4 Round-Robin with Tokens (RR/Token)

This protocol uses tokens to resolve receiver collisions. There are N tokens, one for each destination node. A token may be either available or in use. A node can only transmit to a destination node j, if it captures the j-th token. The transmit queues at each node are served in a Round-Robin manner. The operation of the *Round-Robin with Tokens* (RR/Token) protocol is as follows.

At the transmitter side, node i monitors each received control frame. If it finds an available token, it removes it from the control frame, and puts it in its FIFO token queue. Node i also serves the transmit queues in the arrival order of tokens: if the first token in the token queue is token j, node i first checks whether transmit queue j is eligible for service. If not, node i releases token j and proceeds with the next token in the queue. Otherwise, node i constructs the burst to node j, writes the burst information in the next control frame, and sends it after a delay equal to the offset value. Once the burst transmission is complete, node i releases token j to the next control frame. It then proceeds to serve the transmit queue corresponding to the next token in the token queue. Since every node has a FIFO token queue, the order in which tokens circulate around the ring is fixed. Recall that there are only N tokens, one for each destination node. Therefore, transmit queues are served in a Round-Robin manner.

At the receiver side, node i checks each incoming control frame for any control slot indicating a burst transmission to this node. If such a control slot is found, node i instructs its receiver to tune to the appropriate home wavelength for receiving the burst.

RR/Token is a receiver collision free protocol, since there can be at most one burst transmission arriving at a destination node at any time.

4 Numerical Results

We used simulation to compare the protocols described in the previous section. For each of the four protocols RR/R, RR/P, RR/NP, and RR/Token, we consider two variants: one in which the offset calculation is based on ODD, using expression (2), and one in which the offset calculation is based on JET, using expression (1). In our study we consider a ring network with 10 nodes. We set the (electronic) buffer capacity at each node to 10 MBytes. The distance between two successive nodes in the ring is taken to be 5 Km. We assume that the control wavelength runs at 622 Mbps, while each burst wavelength runs at 2.5 Gbps. Each control slot in a control frame is 100 bytes long regardless of the protocol used in the ring. The processing time of a control frame at both the intermediate ($T_i^{(p)}$) and destination nodes ($T_d^{(p)}$) is set to be 10 slot times, and the switch setup time at the destination nodes $T_d^{(s)}$ is 1 μs.

We model the packet arrival process to each node by a modified Interrupted Poisson Process (IPP) [7]. This modified IPP is an ON/OFF process, where both the ON and the OFF periods are exponentially distributed. Packets arrive back to back during the ON period at the rate of 2.5 Gbps. No packets arrive during the OFF period. The packet size is assumed to follow a truncated exponential distribution with an average size of 500 bytes and a maximum size of 5000 bytes. We use the squared coefficient of variation, C^2, of the packet inter-arrival time to measure the burstiness of the arrival process. C^2 is defined as the ratio of the variance of the packet inter-arrival time divided by the squared mean of the packet inter-arrival time. The arrival process is completely characterized by the C^2 and the average arrival packet rate. In all simulations, we set C^2 to 20, and vary the average arrival rate. Packets arriving at a node are assigned a destination node following the uniform distribution.

Effect of Average Arrival Rate. We first investigate the performance of the four protocols when the calculation of the offset is based on ODD. We consider five performance measures: throughput, loss, delay, fairness, and buffer requirement. Since each node is fed with the same arrival process, the average arrival rate we refer to is the average arrival rate to a single node. We set `MaxBurstSize` to 112 Kbytes and `MinBurstSize` to 16 Kbytes.

Figure 4 plots the mean node throughput against the average arrival rate for all four protocols. We observe that RR/Token, a protocol free of receiver collisions, achieves the highest throughput. Among the three protocols in which

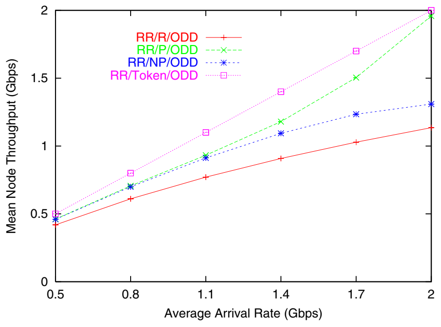

Fig. 4. Mean node throughput

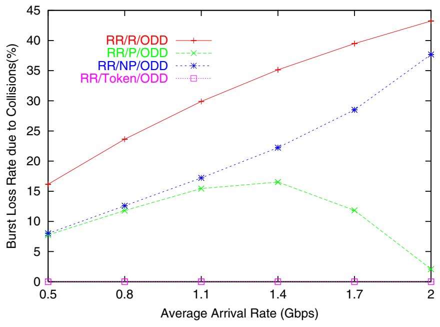

Fig. 5. Burst loss due to collisions

receiver collisions are possible, RR/P achieves the highest throughput, followed by RR/NP and RR/R.

We distinguish between two types of loss. First, packets arriving to find a full buffer at the source node are dropped. In our experiments, we observed that only RR/Token has a 0.01% packet loss rate due to buffer overflow, when the average arrival rate is 2.0 Gbps. The second type of loss occurs when a burst is dropped at the destination due to a receiver collision. Figure 5 plots the burst loss rate due to receiver collisions versus the average arrival rate. As expected, RR/Token never incurs loss due to receiver collisions. For the other three protocols, RR/P has the least burst loss rate, followed by RR/NP and RR/R.

Next, we give an intuitive explanation of the burst loss plots in Figure 5. The behavior of these plots is related to the C^2 of the burst size. If all other parameters are kept the same, a larger burst size C^2 leads to a larger burst loss rate due to receiver collisions. Figure 6 shows the C^2 of the burst sizes as a function of the average arrival rate. We note that the plots in both Figures 5 and 6 have the same pattern. As the average arrival rate increases, the C^2 of the burst size of RR/R and RR/NP increases, and so does the burst loss rate. For RR/P, however, as the average arrival rate increases, the burst size C^2 first increases, it peaks at 1.4 Gbps, and then it decreases. The burst loss rate follows the same pattern. The reason for the change in the C^2 of burst size is that when the burst size reaches a specific point, the `MaxBurstSize` starts to limit the C^2 of burst size.

From the simulation, we also found that the burst loss rate due to receiver collisions of RR/P depends not only on the C^2 of the burst size, but also on another important parameter, the *EnoughData* probability. Recall that in a node, a transmit queue is not eligible for service unless its size is at least equal to the value of `MinBurstSize`. Therefore, when a node turns to serve a transmit queue, the transmit queue may or may not be eligible for service. The probability that a transmit queue is eligible for service when a node turns to serve it is the *EnoughData* probability. We found that for RR/P, an *EnoughData* probability equal to or very close to one leads to a lower burst loss rate due to

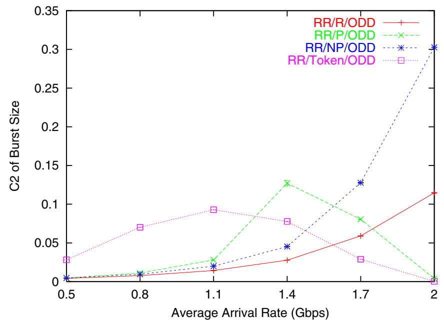

Fig. 6. C^2 of Burst Size

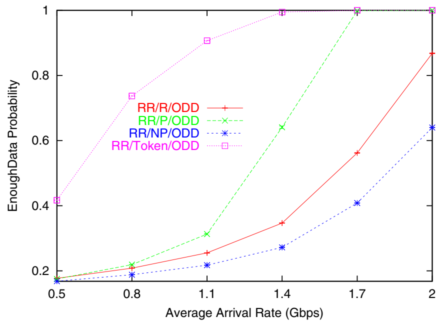

Fig. 7. `EnoughData` probability

receiver collisions than an *EnoughData* probability close to zero. Figure 7 shows the *EnoughData* probability versus the average arrival rate. The *EnoughData* probability of RR/P increases as the average arrival rate increases, reaching 1 when the average arrival rate is 1.7 Gbps.

Figure 8 plots the mean packet delay, including the queueing and propagation delay, versus the average arrival rate. The RR/R protocol has the least delay, followed by RR/NP, RR/P and RR/Token. We observe that, as the average arrival rate increases, the mean packet delay in all protocols first decreases, and then it increases. This behavior is due to the fact that, when the traffic intensity is low, the time for a transmit queue to reach the `MinBurstSize` accounts for the major part of the packet delay. Therefore, as the average arrival rate increases, the time for a transmit queue to reach `MinBurstSize` decreases, which causes the mean packet delay to decrease. The 95% percentile packet delay was also calculated in the simulation. Since the plot trend is the same as that of the mean packet delay, the figure is not shown here.

Let us now compare the four protocols in terms of fairness. We distinguish two types of fairness, namely, throughput fairness and delay fairness. We define the *throughput fairness index of a node i* as

$$\text{Throughput Fairness Index of Node } i = \left(\sum_{j=1, j \neq i}^{10} (H_{ij} - \overline{H_i})^2 \right) \times \frac{1}{\overline{H_i}^2} \quad (3)$$

where H_{ij} is the throughput from node i to node j, i.e., the average number of bits transmitted by node i and received by node j in a unit time, and $\overline{H_i} = (\sum_{j=1, j \neq i}^{10} H_{ij})/9$. We then define the *throughput fairness index of a protocol* as the average of the throughput fairness indices of all nodes. According to this definition, the smaller the throughput fairness index of a protocol, the better the throughput fairness of the protocol.

Figure 9 shows the throughput fairness index of the four protocols versus the average arrival rate. We observe that RR/R and RR/Token have values very close to zero, meaning that they are throughput fair protocols. We have

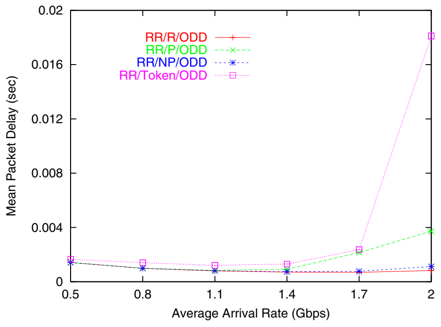

Fig. 8. Mean packet delay

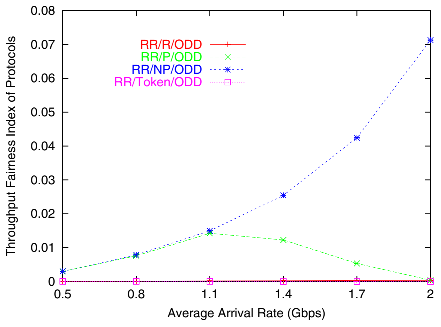

Fig. 9. Throughput fairness index

also computed the throughput from each node to each other node in the ring (these results are not shown here). We have observed that both the RR/NP and RR/P protocols provide better throughput to nodes closer to the source than to nodes far away. This follows directly from the operation of RR/NP and RR/P described in Section 3.3 and 3.2.

The second type of fairness we consider is related to delay. The *delay fairness index of a node i* is defined as

$$\text{Delay Fairness Index of Node } i \;=\; \left(\sum_{j=1, j\neq i}^{10} (W_{ij} \;-\; \overline{W_i})^2 \right) \times \frac{1}{\overline{W_i}^2} \qquad (4)$$

where W_{ij} is the mean queueing delay of packets in transmit queue j of node i, and $\overline{W_i} = (\sum_{j=1,j\neq i}^{10} W_{ij})/9$. We also define the *delay fairness index of a protocol* as the average of the delay fairness indices of all nodes. (Note that in defining the fairness index we use the queueing delay only, not the total delay, since the latter includes the propagation delay which depends on the destination node). According to this definition, the smaller the delay fairness index of a protocol, the better the delay fairness of the protocol. Specifically, if the delay fairness index of a protocol is zero, the protocol is perfectly fair since the queueing delay of a packet is insensitive to the source and destination of the packet. For unfair protocols, access to the burst wavelengths may depend on factors such as the relative position of the source and destination nodes in the ring. In this case, some transmit queues may take longer to serve than others, increasing the queueing delay of the respective packets relative to others, and thus, increasing the delay fairness index of the node and protocol.

Figure 10 shows the delay fairness index of the four protocols versus the average arrival rate. We observe that only RR/R has delay fairness index values very close to zero, meaning that it is the only fair protocol in terms of delay. We have also computed the mean packet queueing delay of each transmit at all ring nodes for the four protocols (these results are omitted due to space limitations). We have observed that the RR/NP protocol provides better delay access to wavelengths of nodes far away than to wavelengths of nodes close to the source of a packet, and RR/P and RR/Token do not always provide the best or worst delay access to a specific node.

Overall, the RR/Token protocol achieves the highest mean throughput, followed by the RR/P, RR/NP and RR/R protocols. RR/R has the smallest mean packet delay, followed by RR/NP, RR/P and RR/Token. RR/R also requires the smallest mean buffer requirement, followed by RR/NP, RR/P and RR/Token. The burst loss rate due to receiver collisions for the protocols which are not free of receiver collisions depends on the C^2 of the burst size. The burst loss rate of RR/P also depends on the *EnoughData* probability. Only RR/R is a delay fair protocol, while both RR/R and RR/Token are throughput fair protocols.

Effect of `MaxBurstSize`. We also varied the value of `MaxBurstSize` from 32 Kbytes to 112 Kbytes with an increment of 16 Kbytes (not shown here). `Min-BurstSize` was set to 16 KBytes, and the average arrival rate to 1.7 Gbps. Simulation results showed that an increase in `MaxBurstSize` leads to an increase in the C^2 of the burst size and to a small change in the *EnoughData* probability; this leads to an increase in the burst loss rate due to receiver collisions, and to a decrease in the throughput of RR/R, RR/NP, and RR/P. However, the decrease in the throughput of RR/R and RR/NP is very small. RR/Token requires a large `MaxBurstSize` so that no packet will be lost due to buffer overflow. Only a very small `MaxBurstSize` could lead to a much longer delay under RR/P and RR/Token. For the other protocols, the change in the mean packet delay because of changes in the `MaxBurstSize` is minimal.

Effect of `MinBurstSize`. We varied `MinBurstSize` from 16 Kbytes to 96 Kbytes while keeping the `MaxBurstSize` at 112 KBytes, and the average arrival rate at 1.7 Gbps. Simulation results showed that an increase in `MinBurstSize` leads to a decrease in the C^2 of the burst size and a decrease in the *EnoughData* probability. For RR/R and RR/NP, the decrease in the C^2 of the burst size leads to a small decrease in the burst loss rate due to collisions, which finally leads to a small increase in the mean node throughout. However, for RR/P, a big decrease in the *EnoughData* probability leads to an increase in the burst loss rate due to receiver collisions, which finally leads to a decrease in the mean node throughout. Changes in `MinBurstSize` do not lead to any change in the mean node throughput of RR/Token. Increases in `MinBurstSize` also lead to increases in the mean packet delay of all protocols.

JET vs. ODD. We now focus on the difference between the JET and ODD offset calculations. Due to space limitations, we only consider two protocols: RR/Token and RR/R. Simulation experiments were carried out with the same parameters as above. The results showed that, compared to ODD, JET leads to a longer mean packet delay for all protocols (see Figure 11), which in turn leads to a larger mean buffer requirement, and to a larger packet loss rate due to buffer overflow. Therefore, as a receiver collision free protocol, RR/Token has a lower mean node throughput with JET than with ODD. Moreover, JET naturally leads to delay unfair protocols, but does not change the throughput fairness property of the protocols.

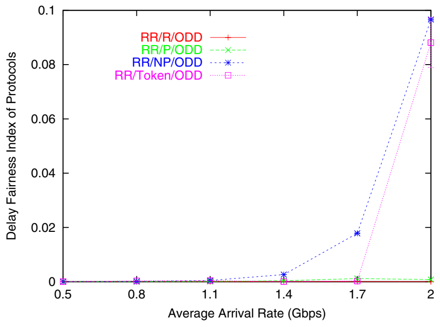

Fig. 10. Delay fairness index

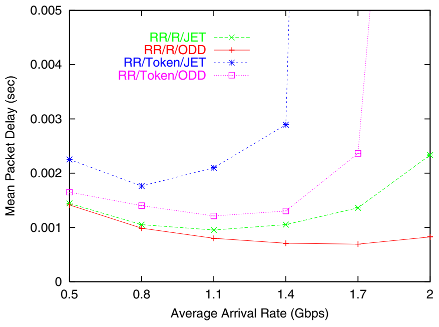

Fig. 11. Mean packet delay

The effect of `MaxBurstSize` was also investigated. The results showed that all protocols are more sensitive to `MaxBurstSize` with JET than with ODD. A much larger `MaxBurstSize` is required in JET than in ODD, in order to get a higher mean node throughput and lower mean packet delay. Results also showed that both ODD and JET are not very sensitive to `MinBurstSize`. As the `MinBurstSize` increases, for RR/R, there is no big difference between ODD and JET. But for RR/Token, ODD is always much better than JET in both the mean node throughput and the mean packet delay.

5 Concluding Remarks

We described a WDM metro ring architecture with optical burst switching. Several access protocols were proposed and their performance was analyzed by simulation.

References

1. L. Xu, H. G. Perros, and G. N. Rouskas. Techniques for optical packet switching and optical burst switching. *IEEE Communications*, 39(1):136–142, January 2001.
2. I. Baldine, G. N. Rouskas, H. G. Perros, and D. Stevenson. `JumpStart`: A just-in-time signaling architecture for WDM burst-switched networks. *IEEE Communications*, 40(2):82–89, February 2002.
3. C. Qiao and M. Yoo. Optical burst switching (OBS)-A new paradigm for an optical Internet. *Journal of High Speed Networks*, 8(1):69–84, January 1999.
4. J. S. Turner. Terabit burst switching. *J. High Speed Networks*, 8(1):3–16, 1999.
5. S. Verma, H. Chaskar, and R. Ravikanth. Optical burst switching: a viable solution for terabit IP backbone. *IEEE Network*, pages 48–53, November/December 2000.
6. Y. Xiong, M. Vandenhoute, and H.C. Cankaya. Control architecture in optical burst-switched WDM networks. *IEEE JSAC*, 18(10):1838–1851, October 2000.
7. W. Fischer and K. Meier-Hellstern. The markov-modulated poisson process (MMPP) cookbook. *Performance Evaluation*, 18:149–171, 1992.