Optical Switching and Networking 10 (2013) 132–148

Contents lists available at SciVerse ScienceDirect

Optical Switching and Networking

journal homepage: www.elsevier.com/locate/osn

New assembly techniques and fast reservation protocols for optical burst switched networks based on traffic prediction

Kyriaki Seklou, Angeliki Sideri, Panagiotis Kokkinos*, Emmanouel Varvarigos

Department of Computer Engineering and Informatics, University of Patras and Research Academic Computer Technology Institute, Patras, Greece

ARTICLE INFO

Article history:
Received 24 November 2009
Received in revised form
24 May 2012
Accepted 4 August 2012
Available online 4 September 2012

Keywords:
Optical burst switched networks
Burst assembly algorithms
Reservation protocols
Traffic prediction

ABSTRACT

We propose new burst assembly schemes and fast reservation (FR) protocols for Optical Burst Switched (OBS) networks that are based on traffic prediction. The burst assembly schemes aim at minimizing (for a given burst size) the average delay of the packets incurred during the burst assembly process, while the fast reservation protocols aim at further reducing the end-to-end delay of the data bursts. The burst assembly techniques use a linear prediction filter to estimate the number of packet arrivals at the ingress node in the following interval, and launch a new burst into the network when a certain criterion, different for each proposed scheme, is met. The fast reservation protocols use prediction filters to estimate the expected length of the burst and the time needed for the burst assembly process to complete. A Burst Header Packet (BHP) packet carrying these estimates is sent before the burst is completed, in order to reserve bandwidth at intermediate nodes for the time interval the burst is expected to pass from these nodes. Reducing the packet aggregation delay and the time required to perform the reservations, reduces the total time needed for a packet to be transported over an OBS network and is especially important for real-time applications. We evaluate the performance of the proposed burst assembly schemes and show that a number of them outperform the previously proposed timer-based, length-based and average delay-based burst assembly schemes. We also look at the performance of the fast reservation (FR) protocols in terms of the probability of successfully establishing the reservations required to transport the burst.

1. Introduction

Optical Burst Switching (OBS) [1] is considered a promising technology for implementing the next generation optical Internet, required to cope with the rapid growth of Internet traffic and the increased deployment of new services (e.g., VoIP, video on demand, cloud computing, digital repositories, data centers). OBS aims at making efficient utilization of the network bandwidth, creating a network infrastructure that is configurable and versatile at the burst level, so as to handle the bursty traffic patterns generated by these services. In circuit switching static optical circuits are established that may not be used most of the time, leading to waste of network resources, while it also requires the aggregation of microflows into circuits, meaning that fine granularity and control over the Quality of Service (QoS) of individual microflows is lost. An ideal network of infinite bandwidth, would most probably use circuit switching instead of OBS, where circuits would be established between any source–destination pair, however such a network does not exists. Additionally all-optical packet switching technology is not yet mature, since packet contention and buffering in the optical domain are yet to be resolved, making OBS easier to implement in practice.

* Corresponding author.
E-mail address: kokkinop@ceid.upatras.gr (P. Kokkinos).

1573-4277/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.osn.2012.08.002

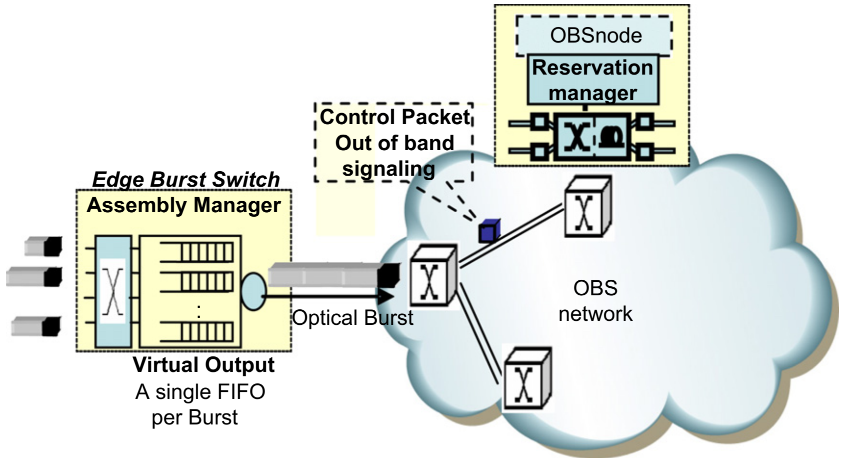

Fig. 1. The architecture of a typical OBS network: (i) the assembly manager at the edge node creates the burst, (ii) a control packet is sent over a control channel in the OBS network reserving the necessary resources, and (iii) the burst completes and is then switched all optically in the network.

The two main ideas in Optical Burst Switching are the assembly of variable sized data packets into bursts that are switched using a single label, and the decoupling of the transmission of the control header from the transmission of the data payload.

Fig. 1 highlights the architecture of a typical OBS network, consisting of a cloud of optical core routers, organized as a mesh, with edge (ingress and egress) routers at the edges of the cloud. During the burst assembly process, multiple packets (IP packets, ATM cells, Ethernet frames etc.) are assembled into big containers (data bursts) at the network ingress. Typically, an ingress edge router maintains a separate (virtual) queue for each Forwarding Equivalent Class (or FEC), defined by the destination and QoS parameters. When a burst assembly threshold is reached, the burst is transmitted over the network. For each data burst, a Burst Header Packet (BHP) containing routing and scheduling information is transmitted ahead of the burst, reserving the required bandwidth and configuring the switches/nodes along the path. The BHP is transmitted over a control channel and is processed electronically. On the other hand the burst is switched throughout the network all-optically through a data channel.

The separation between control and data maintains data transparency and leads to a better synergy of mature electronic technologies (which process the BHP) and advanced optical technologies (which handle the data burst). Finally, at the egress nodes the opposite operation is performed, where bursts are disassembled back to packets.

The end-to-end delay of a burst over an OBS network mainly consists of four components: (i) the burst assembly delay at the ingress node, (ii) the path setup delay caused by the BHP, (iii) the burst transmission time, and (iv) the propagation delay in the core network. The two last delay components depend on the path selected and the available bandwidth on the path and cannot be reduced through clever design of the signaling protocols. This work focuses on the first two delay components, and consists of two main contributions. First, it proposes new burst assembly algorithms that minimize the burst assembly delay, for a given average size of the bursts produced. Second, it uses pipelining techniques to reduce the combined duration of the burst assembly and path setup time and the overall end-to-end delay. The end-to-end delay is a crucial QoS parameter for a number of applications, while the burst drop probability is another important QoS parameter of interest in this work. Finally, the size of the bursts produced is also important in determining the control overhead posed on the network and the efficient use of the available bandwidth.

A number of burst assembly schemes have appeared in the literature, including the time-based algorithm (abbreviated T_{MAX} algorithm) and the length-based algorithm (abbreviated L_{MAX} algorithm) [2,3]. In the time-based algorithm, a time counter is started any time a packet arrives at an empty ingress (FEC) queue, and the burst is completed when the timer reaches the threshold T_{MAX}. In the length-based method, the threshold specifies the number of packets to be aggregated into a burst, or the size of the burst in bytes if the packets are of variable size. Once the threshold L_{MAX} (in fixed size packets, bytes, etc) is reached, the burst is created and sent into the network after an appropriate offset time. The time-based algorithm limits the average burst assembly delay but may generate very small bursts (the authors in [4] provide a quantitative measure of what small or not burst sizes are), while the opposite occurs for the length-based algorithm. For this reason, hybrid schemes [5,6] have also been proposed. An average delay-based algorithm (abbreviated T_{AVE} algorithm) was also introduced in [7] that aims at controlling the average burst assembly delay by letting out the bursts, the moment the average delay of the packets that comprise it reaches a threshold T_{AVE}. This method guarantees a desired average burst assembly delay, and also tends to minimize packet delay jitter, which is particularly important for TCP performance. Relatively, recently adaptive burst assembly algorithms that use the congestion window sizes of TCP flows have also appeared [8,9].

In this work, the proposed schemes and protocols attempt to achieve an optimal trade-off between the average burst assembly delay and the average burst size, using traffic prediction. In Section 2 we propose and evaluate several novel burst assembly schemes that use traffic prediction in order to maximize the average length of the bursts produced for a given average burst assembly delay, or, alternatively, to minimize the average burst assembly delay for a given average burst length. Prediction of traffic characteristics has previously been examined in [10–12]. In a more recent work [13], the authors propose a burst assembly process using traffic predictions, based on intimate flow-level knowledge of TCP traffic. In this work, traffic prediction is used in order to estimate the number of packets that will arrive at the assembly queue in the near future and determine if it would be beneficial for the burst assembly process to wait for these packets, or the burst should be sent immediately.

We find that two of the proposed schemes improve burst assembly efficiency over the previously proposed schemes, by reducing the average burst assembly delay by up to 33% for a given size of the bursts produced, compared to previous schemes.

Following the introduction of the new burst assembly schemes, we turn our attention to signaling protocols for connection establishment and resource reservation in OBS networks, to obtain fast reservation (FR) protocols, so as to reduce the second delay component of OBS networks, that is the path setup delay. A number of signaling protocols for OBS networks have been proposed so far and can be categorized into two main classes: "tell-and-wait" (TAW) and "tell-and-go" (TAG). While the former features a two-way reservation process, the latter uses one-way signaling that releases the burst without waiting for the confirmation of the successful establishment of the path; instead it waits only for a fixed time period equal to t_0 (time-offset). Thus, in TAW schemes reservation requests may be blocked, but bursts are guaranteed to arrive at their destination once they enter the network, while in TAG schemes bursts may be dropped at the core nodes, since resources are not reserved for them in advance. An example of a TAW protocol is the Efficient Reservation Virtual Circuit (ERVC) protocol proposed in [14], while other research efforts include the Wavelength Routed-OBS (WR-OBS) [15]and Efficient Burst Reservation Protocol (EBRP) [16]. TAG reservation schemes include the Ready-to-Go Virtual Circuit (RGVC) [17], Horizon [18], Just Enough Time (JET) [19,20] and Just In Time (JIT) [21,22] protocols. A detailed analysis of the JET, JIT and Horizon reservation schemes is presented in [23]. Enhanced versions of these protocols have also been proposed, such as in [24]. Also, in [25] the authors propose the distribution of the BHP information to all network nodes, so as to improve network efficiency.

In Section 3 we extend the work in [12] and present a general formulation for combining fast reservation (FR) schemes with the burst assembly algorithms introduced earlier in this paper (T_{MAX}, L_{MAX}, and T_{AVE}) using one or two linear prediction filters [26]. Moreover, the proposed FR scheme combined with T_{AVE} can also be used with the novel burst assembly algorithms presented in Section 2. In [12] the authors evaluate the use of a linear prediction filter along with the T_{MAX} burst assembly algorithm. The prediction filters estimate the length of the burst and/or the time needed for the burst assembly process to complete and this make possible the pipelining of the reservation and the burst assembly processes. The goal is (i) to reduce the end-to-end delay of a data burst, by minimizing the burst pretransmission time (the time-offset), while (ii) using bandwidth efficiently by reserving resources for a duration that is close to the minimum possible. The performance results indicate that the method first proposed in [12] can also be used along with L_{MAX} and T_{AVE} burst assembly algorithms.

The remainder of the paper is organized as follows. In Section 2, we present four new burst assembly schemes. In Section 3, we present a general formulation for combining fast reservation (FR) schemes that use predictions with burst assembly algorithms. In Section 4, we describe the prediction filters we use, while in Section 5 we present the performance results obtained for the proposed burst assembly schemes (Section 5.1) and for the fast reservation (FR) protocols (Section 5.2). Finally in Section 6, the paper's conclusions are presented.

2. Burst assembly algorithms

In this section we focus on the burst assembly process, and propose burst assembly algorithms that try to minimize the average burst assembly delay, for a given average size of the bursts produced. Generally, the burst assembly process at an edge node starts with the arrival of the first packet at an empty queue and continues until a predefined threshold is reached. Different assembly strategies define this threshold differently , and try to balance between two objectives: the burst assembly delay and the size of the bursts produced. Short burst assembly delays and large burst sizes are desirable, in order to reduce, respectively, the total end-to-end delay and the number of bursts along with the processing overhead they pose on the core nodes. These objectives, however, contradict each other (Fig. 2), since increasing the burst size also increases the burst assembly delay. A burst assembly algorithm, therefore, should be judged based on how well it performs with respect to one of these two performance metrics of interest, for a given value of the other performance metric. In Fig. 2, the burst assembly algorithm used determines the curve that relates the average burst size to the average burst assembly delay. Given a burst assembly algorithm, choosing the desired

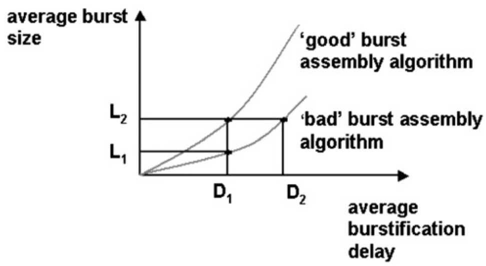

Fig. 2. Performance of a burst assembly algorithm. Based on the requirements of the user or of the application such an algorithm can be characterized as "good" or as "bad". For example, a "good" algorithm produces bursts of larger average size, for a given average burst assembly delay (alternatively, they result in smaller average burst assembly delay for a given average size of the bursts produced).

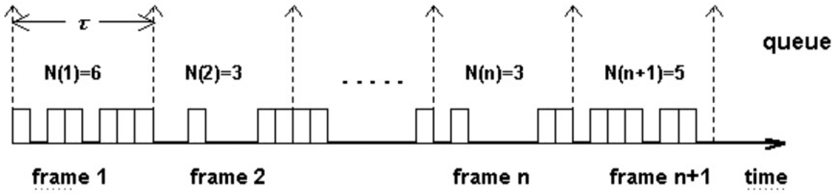

Fig. 3. Time frame structure. At the end of each frame n the algorithm decides if it should send out the burst immediately, or it should wait for another frame. $N(n)$ is the number of packet arrivals during the nth frame.

balance between the burst assembly delay and the burst size (that is, the exact point on the curve of Fig. 2 at which the system operates) depends on the Quality of Service (QoS) requirements of the users, and the processing and buffering capabilities of the backbone nodes.

In the proposed schemes, we assume that the time axis is divided into time frames of equal duration τ (see Fig. 3). During a frame, an edge OBS node assembles the packets arriving with the same destination address and the same QoS requirements (that is, the same FEC) into a burst. We denote by $N(n)$ the number of packet arrivals during the nth frame. At the end of each frame, a decision is taken about whether the burst should be sent out immediately and the assembly of a new burst should start, or the edge node should wait for another frame in order to include more packets in the current burst. This decision is taken by using a linear prediction filter (described in Section 4) to estimate the expected number $\hat{N}(n+1)$ of packet arrivals in the following frame $n+1$, and checking if a specific criterion (different for each algorithm proposed) is fulfilled. This criterion tries to quantify if the increase in the burst length expected by waiting for an extra frame is significant enough to warrant the extra delay that will be incurred.

The following subsections describe the proposed burst assembly algorithms, while the corresponding performance results and comparison between the schemes are deferred until Section 5.1.

2.1. Fixed additional packets threshold algorithm (N_{MIN} algorithm)

In this scheme, we define a lower bound N_{MIN} on the number of future arrivals above which we decide to wait for an extra frame before completing the burst. At the end of frame n, the estimate $\hat{N}(n+1)$ produced by the predictor is compared to the threshold N_{MIN}, and if it is smaller than that, the burst leaves the queue immediately, limiting in this way the burst assembly delay; otherwise it waits for another frame to be completed, at the end of which the same procedure is repeated. Therefore, the burst is sent out at the end of the nth frame if and only if

$$\hat{N}(n+1) < \text{N}_{\text{MIN}}. \tag{1}$$

2.2. Proportional additional packets threshold algorithm (aL algorithm)

In this proposed scheme, instead of using a fixed threshold N_{MIN}, a fraction of the current burst length is used as the threshold. If α is the multiplicative parameter, where $\alpha \in (0{,}1]$, the burst is completed at the end of frame n, if and only if

$$\hat{N}(n+1) < a \cdot L(n), \tag{2}$$

where $L(n)$ is the burst length at the end of the nth frame, and $\hat{N}(n+1)$ is the predictor's estimate for the number of packets expected during the following frame $n+1$.

2.3. Average delay threshold algorithm (T_A algorithm)

This method tries to improve on the average-delay-based algorithm proposed in [7], which computes a running average of the packet burst assembly delay and lets out the burst, the moment the average delay of the packets that comprise it reaches a threshold T_{AVE}. The algorithm in [7] has two drawbacks: (a) computing the running average introduces considerable processing overhead, and (b) bursts may not be sent out at the optimal time, since the running average is non-monotonic in time and could decrease in the future due to new packet arrivals. If the latter (b) scenario is repeated several times then it is possible that the burst assembly delay is increased beyond acceptable levels. The T_A algorithm addresses these drawbacks using traffic prediction. At the end of each frame, it estimates the average burst assembly delay we expect to have at the end of the following frame, and launches the burst if this estimate exceeds some threshold value T_A.

The Average Packet Delay $PD(n)$ of the packets in the burst assembly queue at the end of frame n is defined as

$$PD(n) = \frac{\sum_{i=1}^{L(n)} T_i(n)}{L(n)} (\text{in } s/\text{packet}),$$

where $L(n)$ is the burst size (in packets) at the end of frame n, $T_i(n) = n\tau - t_i$ is the delay of the ith packet from the moment it enters the queue until the end of nth frame, τ is the duration of the frame, and t_i is the arrival time of the ith packet. Alternatively and more easily, we can compute $PD(n)$ using the recursion:

$$PD(n) = \frac{L(n-1)PD(n-1) + L(n-1)\tau + \sum_{i=1}^{N(n)} T_i(n)}{L(n-1) + N(n)}, \tag{3}$$

where $N(n)$ is the number of packet arrivals during frame n. If a burst was sent out at the end of the $(n-1)$th frame, we take $L(n-1) = 0$ in Eq. (3).

To obtain an estimate $\hat{PD}(n+1)$ of the Average Packet Delay at the end of frame $n+1$ we assume that the $\hat{N}(n+1)$ packets estimated by the predictor to arrive by the end of frame $n+1$ will have an average delay of $\tau/2$. Using Eq. (3), the estimated Average Packet Delay $P\hat{D}(n+1)$ at the end of frame $n+1$ is

$$P\hat{D}(n+1)=\frac{L(n)PD(n)+L(n)\tau+\hat{N}(n+1)\tau/2}{L(n)+\hat{N}(n+1)}. \tag{4}$$

A burst is completed at the end of the nth frame if and only if

$$P\hat{D}(n+1)>T_A, \tag{5}$$

where T_A is the predefined threshold value.

2.4. Average delay to burst size ratio improvement algorithm (L_{MIN} algorithm)

The proposed Average Delay to Burst Size Ratio Improvement algorithm is based on the Average Packet Burst Assembly Delay to Burst Size ratio (DBR) defined as

$$DBR=\frac{\text{Average Packet Burst Assembly Delay}}{\text{Average Burst Size}}(s/\text{packet})$$

The algorithm with threshold L_{MIN} (abbreviated L_{MIN} algorithm) uses traffic prediction to compute an estimate $D\hat{B}R(n+1)$ of DBR at the end of frame $n+1$, and decides that the burst is completed, if this estimate is worse (larger) than the current value $DBR(n)$. The average burst assembly delay to burst size ratio $DBR(n)$ at the end of frame n is defined as:

$$DBR(n)=\frac{PD(n)}{L(n)}=\frac{\sum_{i=1}^{L(n)}T_i(n)}{L^2(n)}.$$

Alternatively, and more easily, $DBR(n)$ can be found recursively as:

$$DBR(n)=\frac{L(n-1)PD(n-1)+L(n-1)\tau+\sum_{i=1}^{N(n)}T_i(n)}{(L(n-1)+N(n))^2}. \tag{6}$$

The Estimated Average Packet Burst Assembly Delay to Burst Size ratio $D\hat{B}R(n+1)$ at the end of frame $n+1$ is:

$$D\hat{B}R(n+1)=\frac{L(n)\cdot PD(n)+L(n)\cdot\tau+\hat{N}(n+1)\cdot\tau/2}{(L(n)+\hat{N}(n+1))^2}. \tag{7}$$

The algorithm decides that a burst is completed and should be sent out at the end of frame n if and only if:

$$D\hat{B}R(n+1)>DBR(n), \tag{8}$$

and

$$L(n)>L_{MIN}. \tag{9}$$

During the first frames that follow a burst assembly completion, there is a great likelihood that the right term of Eq. (8) will be quite small, making it difficult to fulfill. The threshold L_{MIN} is used as a lower bound on the length of the bursts, and also makes the algorithm parametric (as with the previous algorithms examined) so that the desired trade-off between the average burst size and the average packet Burst Assembly delay can be obtained.

Simulation results on the performance of the preceding burst assembly algorithms are presented in Section 5.1. In what follows, we turn our attentions to signaling protocols for reducing the second delay component incurred in OBS networks, which is the path setup delay.

3. Fast reservation protocols

In this section, we look into fast reservation (FR) protocols that can be used with the T_{MAX}, L_{MAX}, T_{AVE} or any of the other burst assembly algorithms introduced in Section 2, to reduce the combined duration of the burst assembly and path setup time, further reducing the overall end-to-end delay. Such a scheme was first presented in [12], where a fast reservation protocol using the T_{MAX} assembly scheme and a prediction filter was proposed. In this section we present a general formulation for combining FR schemes with the T_{MAX}, L_{MAX}, and T_{AVE} burst assembly algorithms, using one or two prediction filters to estimate the burst length and/or the time needed for the burst assembly process. In contrast to most OBS signaling protocols, in the FR schemes the BHP is sent to the core network *before* the burst assembly process is completed, in order to reserve the appropriate resources. Intermediate nodes use the estimated values for the burst length and assembly completion time, instead of the actual values that are not yet known, in order to reserve bandwidth for the intervals the burst is expected to pass from these nodes. Estimating the length of the burst is required in order to reserve the required resources for the right duration for the burst's all-optical transmission. Similarly, estimating the duration of the burst assembly process is required in order to determine the time these reservations should start at the core nodes.

We let $L(k)$ be the size of the kth burst and $D(k)$ be its assembly process duration (Fig. 4). If both were known at the start of a burst assembly, we could start the reservation process at that time, reducing the overall delay. Since $L(k)$ and $D(k)$ are not known in advance, the idea is to start the reservation process before the burst is assembled, using estimates of these variables. In the time-based burst assembly algorithm $D(k)$ is equal to T_{MAX} (therefore, we only estimate L(k)), while in the length-based algorithm $L(k)$ is equal to L_{MAX} (therefore, we then only estimate $D(k)$). In the average delay-based algorithm, as well as in all the burst assembly algorithms of Section 2, both the burst length $L(k)$ and the burst assembly duration $D(k)$ vary and have to be estimated.

3.1. Fast reservation for the T_{AVE} scheme

The signaling used by the FR protocol for the T_{AVE} assembly scheme is illustrated in Fig. 5 (the cases of the burst assembly algorithms of Section 2 are similar, while those of the T_{MAX} algorithm and the L_{MAX} algorithm are simpler): upon the beginning of a new burst assembly period we use two Least Mean Squares (LMS) filters to predict burst related values. Using these predictions, a BHP is sent at the beginning of the burst assembly process to reserve in advance the required resources, instead of waiting for the burst assembly to complete.

Specifically, the first LMS filter is used to predict the length $\hat{L}(k)$ of the kth burst to be formed; this value is included in the BHP and is used to reserve bandwidth for

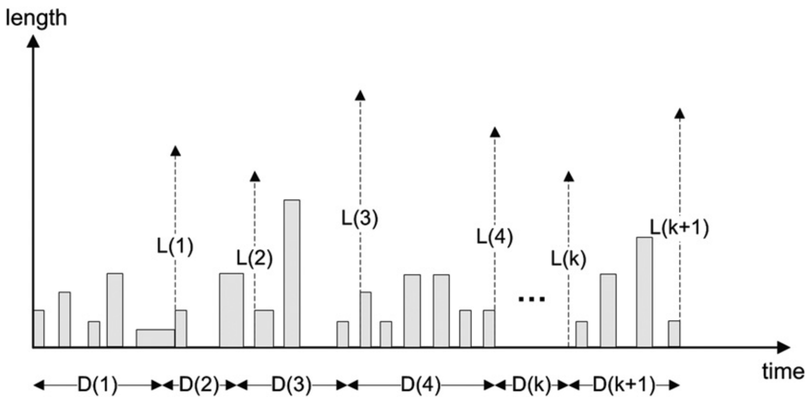

Fig. 4. Prediction of the burst size and burst assembly duration is performed based on the k previous burst lengths and assembly durations. $L(k)$ denotes the size of the kth burst and $D(k)$ is its assembly process duration.

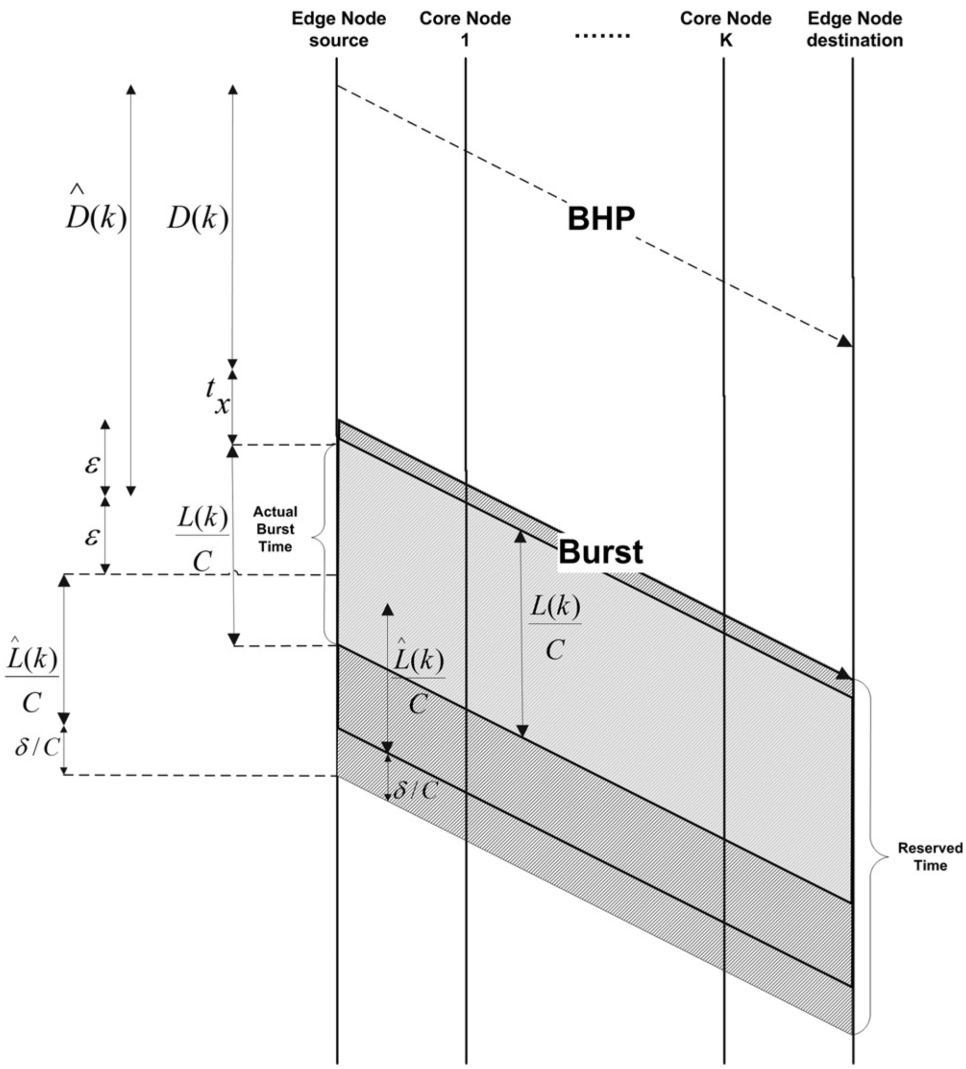

Fig. 5. A successful reservation for the T_{AVE} burst assembly algorithm, using two predictive filters. One filter predicts the burst length $\hat{L}(k)$and the other the burst assembly duration $\hat{D}(k)$. The figure illustrates the time instants at which reservations start and finish at each node of the selected path, and the time intervals the burst actually passes from a node. We should node that the OEO (Optical-Electrical-Optical) conversion that the BHP is experiencing in each node is not depicted in this figure.

a duration close (if the prediction is accurate) to the burst's real transmission time. The second filter produces a prediction of the assembly process duration $\hat{D}(k)$, which is also included in the BHP, and is used to reserve bandwidth at intermediate links starting at the correct (if the prediction is accurate) time instant. To reduce the effects of prediction errors in the burst length, we add a small margin δ in the estimated burst length $\hat{L}(k)$. This is done in order to reduce the probability that bandwidth is reserved for less time than the actual burst duration. No matter how accurate is the filter, the actual length $L(k)$ of will be larger than $\hat{L}(k)$ approximately half of the time, which would be unacceptable; however, $L(k)$ will be smaller than $\hat{L}(k)+\delta$ with high probability if the prediction filter is good and δ is large enough. Similarly, to reduce the effects of prediction errors in the assembly process duration, we subtract a small margin ε from the estimated duration $\hat{D}(k)$. This is because $\hat{D}(k)$ is used to calculate the time at which reservations at intermediate links should start, and in case of uncertainty, it is safer to start reservations a little earlier than the predicted time. By using these safety margins, the reservation starts earlier than the expected time by ε and finishes later than the expected time by $\varepsilon+\delta/C$, where C is the reserved bandwidth. This way there is high probability that the burst will find capacity already reserved for it when it arrives at a node. Therefore, bandwidth is reserved at each intermediate node for the time period:

$$\left[\hat{D}(k)-\varepsilon,\ \hat{D}(k)+\varepsilon+\frac{\hat{L}(k)+\delta}{C}\right], \tag{10}$$

where times are relative to the arrival time of the BHP at each node. Note that if the estimators of $L(k)$ and $D(k)$ are unbiased, capacity is reserved for a burst for time $2\varepsilon+\delta/C$ more than the minimum required, on the average, which may result in a degradation of network efficiency. However, the inefficiency caused by this is negligible if ε and δ are small.

When burst assembly is completed, the predicted values $\hat{D}(k)$ and $\hat{L}(k)$ are compared with the real values of $D(k)$ and $L(k)$. The ingress node sends the burst after a small (pre-transmission) interval t_x, calculated so as to compensate for predictions errors, as will be described shortly. If the burst is sent after time t_x, the time period the burst actually traverses the network is:

$$\left[D(k)+t_x,\ D(k)+t_x+\frac{L(k)}{C}\right], \tag{11}$$

where $L(k)$ is the burst's actual length and $D(k)$ its assembly duration. For the in advance reservation to be successful, the reservation period must contain the burst's actual transmission period. That is, the reservation at any core node should start before the burst arrives and should finish after the burst's departure. So, based on Eqs. (10) and (11) the following conditions must hold:

$$t_x+D(k)>\hat{D}(k)-\varepsilon, \tag{12a}$$

and

$$t_x+D(k)+\frac{L(k)}{C}<\hat{D}(k)+\varepsilon+\frac{\hat{L}(k)+\delta}{C}. \tag{12b}$$

The pretransmission time t_x can be chosen equal to

$$t_x=\max(\hat{D}(k)-\varepsilon-D(k),0),$$

so as to minimize pre-transmission delay, while always satisfying Eq. (12a). In that case,

$$\Pr(t_x=0)=\Pr(D(k)>\hat{D}(k)-\varepsilon),$$

and the pre-transmission delay will be zero with high probability. To satisfy Eq. (12b) it is sufficient that

$$L(k)<\hat{L}(k)+\delta, \tag{13a}$$

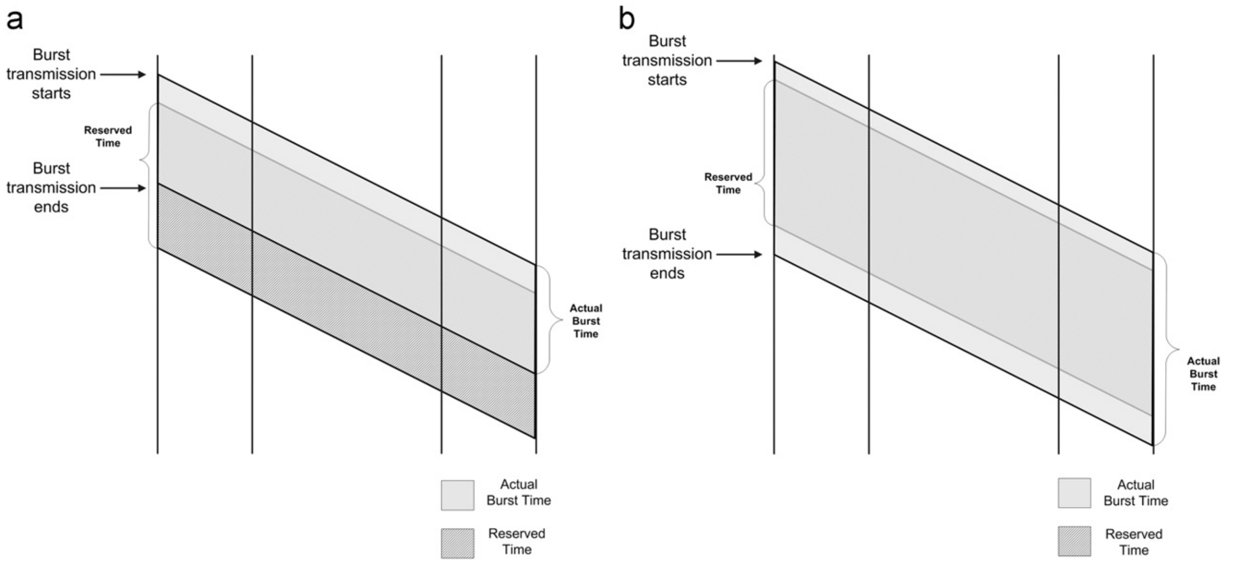

Fig. 6. Failed reservations for the T_{AVE} assembly algorithm. (a) Illustrates the case burst transmission starts earlier than the reservation, and (b) illustrates the case the burst length exceeds the reserved duration.

and

$$D(k) < \hat{D}(k) + \varepsilon, \tag{13b}$$

which will be both valid with high probability when ε and δ are sufficiently large. In that case the reservation will be successful, in the sense that bandwidth will be reserved for a duration that is close to (and larger than) the burst's real transmission duration. Also, the pre-transmission delay will be zero with high probability.

If Eqs. (12a and b) cannot be simultaneously satisfied for any choice of t_x, the transmitted BHP is a failure (Fig. 6). We then have to transmit a new BHP to cancel the old reservation and perform a new one with the actual burst size $L(k)$ and the right reservation starting time. The failed reservation does not result in a burst loss, but only some loss of efficiency in the small interval between the reservation and the cancellation, where the resource remains idle.

In general, the exact implementation of a fast reservation (FR) scheme (as we will also see in Sections 3.2 and 3.3) depends on the burst assembly scheme used and on which parameters (e.g., burst assembly delay or burst length) need to be predicted. Based on this observation, the burst assembly schemes proposed in Section 2 can be combined with the FR scheme used for the T_{AVE} algorithm. By combining this FR scheme with the N_{MIN}, αL, T_A, and L_{MIN} algorithms we jointly minimize the burst assembly delay, for a given average size of the bursts produced and the burst pretransmission time. At the same time we attempt to make efficient use of the available bandwidth both in terms of the final burst size and in terms of the accuracy of the reservations performed.

3.2. Fast reservation for the L_{MAX} scheme

In the case of the L_{MAX} assembly algorithm, the burst length $L(k)$ is fixed and known *a priori*. In that case reservations are performed as in Fig. 5, but with $L(k)=L_{MAX}$ and $\delta=0$. A prediction filter is used to obtain the estimate $\hat{D}(k)$ of the kth burst assembly duration, on which we use a small safety margin ε to compensate for the case the prediction turns out to be larger or smaller than the actual value. The BHP is sent to reserve the necessary resources starting a little earlier than the predicted time, without waiting for the burst assembly to complete. Specifically, the BHP, upon its arrival at a core node, reserves bandwidth C for the time period:

$$\left(\frac{\hat{D}(k)-\varepsilon, \hat{D}(k)+L_{MAX}}{C+\varepsilon}\right), \tag{14}$$

relative to its arrival time at that node. When the burst assembly is completed, the actual assembly duration $D(k)$ is compared to $\hat{D}(k)-\varepsilon$.This comparison is performed in order to ensure that the reservation of the resources in the network starts at the right time. The pre-transmission time is again chosen according to

$$t_x = \max(\hat{D}(k)-\varepsilon-D(k), 0).$$

Provided that

$$D(k) < \hat{D}(k) + \varepsilon, \tag{15}$$

the reservation made by the BHP is successful (if $\hat{D}(k)-\varepsilon < D(k) < \hat{D}(k)+\varepsilon$, we also have $t_x=0$). Otherwise, the BHP is a failure and we transmit a new BHP to cancel the old reservation and perform a new one.

3.3. Fast reservation for the T_{MAX} scheme

In the T_{MAX} assembly algorithm, the burst assembly duration is known *a priori*, since $D(k)=T_{MAX}$. In that case reservations are performed as in Fig. 5 but with $\hat{D}(k)=T_{MAX}$ and $\varepsilon=0$. A filter is used to predict the kth burst length $\hat{L}(k)$, and bandwidth is reserved for time $(\hat{L}(k)+\delta)/C$. When the time threshold T_{MAX} is reached, the burst assembly is completed, and the actual burst length $L(k)$ is compared with the predicted length. If

$$\hat{L}(k) + \delta > L(k), \tag{16}$$

the BHP reserves capacity for enough duration and the reservation is successful. Otherwise, the BHP is a failure and a new BHP is sent to cancel the old reservation and perform a new one for the actual burst size $L(k)$.

3.4. Fast reservation and minimum separation

The transmission of the BHP has to precede the burst transmission by at least a time offset equal to t_0, where parameter t_0 is chosen to account for the extra processing delays the BHP (which is processed electronically) encounters at intermediate nodes when compared with the processing delays encountered by the burst (which is switched all-optically). For example, if t_{el} is the time it takes for a node to process electronically the BHP and t_{ao} is the time it takes for the node to switch (all-optically) a burst from an input to an output port, we can choose

$$t_0 = h \cdot (t_{el} - t_{ao}), \tag{17}$$

where h is the number of hops on the path.

If the estimate $\hat{D}(k)$in the length-or average delay-based burst assembly algorithm is less than t_0, the estimate $\hat{D}(k)$ carried by the BHP in the signaling protocol is replaced by $\max(\hat{D}(k), t0)$. The total burst assembly and pre-transmission delay when a fast reservation (FR) TAG protocol is used is (assuming $t_x > 0$)

$$T_{FR} = \max(t0, D+t_x) = \max(t_0, \hat{D}-\varepsilon), \tag{18}$$

[for the T_{MAX} algorithm, $T_{FR} = \max(T_{MAX}, t_0)$], while if a standard reservation TAG protocol is used, it is

$$T_{SR} = D + t_0. \tag{19}$$

Comparing (18) and (19), the delay reduction achieved through pipelining by the FR protocol becomes evident.

It is natural to assume that the burst assembly thresholds in the T_{MAX}, L_{MAX} and T_{AVE} burst assembly algorithms, and in the burst assembly schemes proposed in Section 2, are chosen so that the average burst assembly duration satisfies $E(D) > t_0$. For example, in the time-based algorithm (T_{MAX}), it is natural to choose the corresponding threshold so as $T_{MAX} > t_0$, since otherwise we could extent the burst assembly period to get larger bursts without any cost in delay. So under these assumptions we can see that the amount of time by which the total end-to-end delay is reduced using an FR protocol, is approximately equal to the time offset t_0.

3.5. Choice of the safety margins δ and ε

The safety margins δ and ε are used to reduce the prediction error effects in the proposed FR schemes. Specifically, we add a small margin δ in the estimated burst length $\hat{L}(k)$, to reduce the probability that bandwidth is reserved for less time than the actual burst duration. We also subtract a small margin ε from the estimated burst assembly duration $\hat{D}(k)$, since $\hat{D}(k)$is used to calculate the starting times of the reservations at intermediate links, and in case of uncertainty, it is safer to start reservations a little earlier than the predicted time, so as to be reasonably sure that the burst will find capacity reserved for it when it arrives at a node.

The values of δ and ε significantly impact the success probability of the BHP reservation (the larger δ and ε are, the larger the probability) and the system costs (the smaller δ and ε are, the smaller the time interval during which capacity is reserved but not used). To obtain a good success probability without substantially increasing system costs, δ is chosen to be a multiple of the root mean square (RMS) of the sample residuals of the LPF,

$$\delta = c_\delta \sqrt{\frac{\sum_{i=1}^{h} e_L{}^2(k-i)}{h}}, \tag{20}$$

where c_δ is a small constant (e.g., 2 or 3), to be referred to as the burst length correction parameter in the rest of the document, $e_L(k)$ is the residual error between the actual and the predicted burst length, and h is the order of the prediction filter used. Similarly, ε is calculated using the corresponding RMS of the residual errors $e_D(k)$ between the actual and the predicted burst assembly durations, according to

$$\varepsilon = c_\varepsilon \sqrt{\frac{\sum_{i=1}^{h} e_D^2(k-i)}{h}}, \tag{21}$$

where c_ε is the duration correction parameter constant.

4. Linear predictor LMS

Predicting network traffic's future characteristics is in general a difficult task. Also, as the authors in [27] mention, how one assesses traffic predictability depends on how one wants to use the prediction results and of course highly depends on the traffic characteristics. For example, traffic prediction demonstrates encouraging potentials when applied to network backbone traces, or aggregate end-system sources. Also, a proper traffic measurement interval or sampling rate has critical effect on prediction. In this work, we assume that short period prediction is possible using a linear prediction filter and focus on the advantages of such a prediction in OBS. However, it is clear that more work is needed on the exact prediction model and input parameters used, which can ensure that such a prediction is valid. A related work is presented in [28], where the authors focus on the parameters of the training-based models used for short period prediction (milliseconds to minutes) of the traffic throughput, showing that this is in general possible.

As mentioned, the Least Mean Square (LMS) filter [29] has been chosen as the linear predictor in this work. This predictor, similar to those used in [11,12], is simple, fast and effective, and has small computational overhead.

The estimate $\hat{N}(n)$ of the number of packet arrivals during the nth frame is obtained as

$$\hat{N}(n) = \sum_{i=1}^{h} w_N(i) N(n-i), \tag{22}$$

where $N(n-i)$ is the number of packet arrivals during the $(n-i)$th frame and h is the length of the filter. The estimate $\hat{L}(n)$ of the length of the nth burst is similarly obtained as

$$\hat{L}(n) = \sum_{i=1}^{h} wL(i) L(n-i), \tag{23}$$

where $L(n-i)$ is the length of burst n–i. Finally, the estimate of the nth burst assembly duration is obtained as

$$\hat{D}(n) = \sum_{i=1}^{h} wD(i) D(n-i), \tag{24}$$

where $D(n-i)$ is the duration of the $(n$–$i)$th burst.

There are a variety of ways to obtain the filter coefficients $w(i)$, $i=1,2,\ldots,h$. In the experiments performed we used the LMS-based recursive LPF that updates the filter coefficients using a simple and efficient algorithm. Specifically, the coefficients for the kth prediction period are obtained according to:

$$w_N(k) = w_N(k-1) + \mu e_N(k-1) N(k-1),$$

$$w_L(k) = w_L(k-1) + \mu e_L(k-1) L(k-1),$$

$$w_D(k) = w_D(k-1) + \mu e_D(k-1) D(k-1),$$

where constant μ is a step-size that affects how quickly the adaptive filter will converge toward the unknown system, $e_N(k-1)$ is the error between the actual and the predicted number of packet arrivals during the $(k$–$1)$th frame, $e_L(k-1)$is the error between the actual and the predicted length of the $(k$–$1)$th burst, and $e_D(k-1)$ is the error between the actual and the predicted duration of the $(k-1)$th burst assembly period. The time complexity for the coefficient calculation of the LMS-based approach is $O(h)$.

5. Performance analysis and simulation results

In this section we present simulation results on the performance of the proposed burst assembly schemes and fast reservation (FR) protocols. In particular, the burst assembly schemes are examined and compared in Section 5.1, while the performance of the FR protocols is discussed in Section 5.2.

5.1. Burst assembly techniques

Using the Matlab environment, we simulated the burst assembly process at an ingress queue in order to evaluate the performance of the N_{MIN}, αL, T_A, and L_{MIN} schemes proposed in Section 2, and compare it to that of the previously proposed T_{AVE}, T_{MAX}, L_{MAX} schemes. We also quantified the impact the choice of the parameters N_{MIN}, αL, T_A and L_{MIN} and of the frame size τ and filter order h have on performance.

It is useful to remind the reader that each of the proposed schemes corresponds to a different Burst Size versus Packet Burst Assembly Delay curve (see the discussion in Section 1 and Fig. 2), while the choice of the parameters involved (N_{MIN}, αL, T_A, L_{MIN}, T_{AVE}, T_{MAX}, L_{MAX}) determines the exact points on each curve the burst assembly process is operating at, that is, the desired trade-off between burst assembly delay and burst size.

5.1.1. Simulation parameters

In the experiments, the arrivals at the ingress queue were obtained from an Exponential-Pareto traffic generating source of rate r bits/s. The traffic source generates superpackets (they can also be viewed as busy periods) with exponentially distributed interarrival times of mean $1/\lambda$ s. The size of each superpacket follows the Pareto distribution with shape parameter β. If a super-packet has size greater than l bytes, which is taken to be the size of the packets used in the network, it is split and sent as a sequence of packets of size l. The time units used for displaying the results are measured in packet slots, where 1 slot$=l/r$ (the transmission time of a packet).

The values of the parameters used in the performed experiments were $\beta=1.2$, $r=1$ Gbps and $l=1500$ bytes. We used $1/\lambda=1.6$ ms or 4.8 ms, corresponding to load utilization factors $p=0.1$ and $p=0.3$. The parameter β determines the Hurst parameter $H=(3-\beta)/2$, which takes values in the interval [0,1] and defines the burstiness of the traffic. The closer the value of H is to 1, the more bursty is the traffic generated.

5.1.2. Predictor performance

The accuracy of the estimations produced by the LMS predictor used in the burst assembly schemes of Section 2 can be assessed by the relative error of the prediction, defined as the inverse of the signal-to-noise ratio:

$$SNR^{-1} = \frac{\sum e^2(k)}{\sum N^2(k)},$$

where $N(k)$ is the actual number of packet arrivals during the k^{th} frame, $\hat{N}(k)$ is its estimated value at the beginning of that frame, and $e(k)=N(k)-\hat{N}(k)$. The results of Fig. 7 examine the dependence of the performance of the LMS predictor on the frame duration τ, the order of the prediction filter h, and the traffic load p. In particular, Fig. 7a shows the way the relative error varies with the frame duration τ for bursty traffic ($H=0.9$). As expected, short frame durations result in smaller values of relative error, since for bursty traffic, the traffic characteristics remain static only for short periods of time. For light traffic, the predictor's performance is worse than it is for heavy traffic. This can also be seen in Fig. 7b, which illustrates the impact that the order of the filter has on relative error. This figure also demonstrates that there is very little improvement when the order of the filter is increased beyond a certain value. This is in agreement with the results in [10], where it was argued that the performance of linear predictors for internet traffic is dominated by short-term correlations, and we do not have to "look deep" into the history of traffic arrivals to obtain a valid estimation. A small order of filter is, therefore, preferable, since it also implies smaller computation overhead. As the frame size τ increases, the relative prediction error remains steady after a certain value ($\tau>0.005$ s) when the traffic is light ($p=0.1$, $p=0.03$), while it worsens slightly for heavier traffic ($p=0.3$).

5.1.3. Comparison between the burst assembly schemes

In this section we compare the average burst size versus average burst assembly delay performance of the proposed burst assembly schemes to that of the previously proposed T_{AVE}, T_{MAX}, L_{MAX} schemes. The results

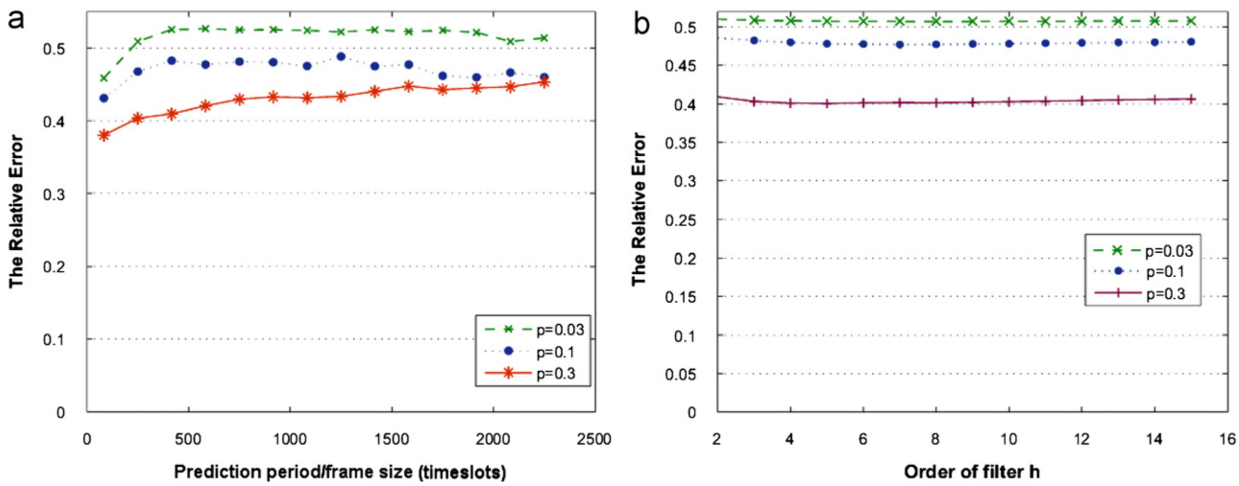

Fig. 7. LMS performance for various loads p and values of: (a) the prediction period τ, and (b) the length h of the predictor.

reported here were obtained for bursty traffic (H=0.9) and varying load utilization factor p. The length h of the LMS predictor was set to 4, while the frame size τ varied depending on the traffic load. The parameters of all the schemes were chosen to produce average burst assembly delays that lie in the same range so that the resulting burst sizes can be compared. Time delays are measured in slots. Figs. 8–10a illustrate the average burst size versus the average packet burst assembly delay when the traffic load is p=0.03, 0.1 and 0.3, respectively. The labels in Figs. 8–10b display the details on the values of the parameters N_{MIN}, T_A, L_{MIN} and α that give the corresponding results.

Figs. 8–10a show that the L_{MAX} algorithm exhibits (as expected) the worst performance for light load (p=0.03), while its performance becomes relatively better for heavier load (p=0.1 and 0.3). The opposite is true for the N_{MIN} algorithm, whose relative performance is worse for heavy traffic, and improves for light traffic. For a given traffic load, the N_{MIN} algorithm exhibits worse relative performance when the parameter N_{MIN} is set at low values so as to produce large bursts. This is because for small values of N_{MIN}, the algorithm cannot well tolerate estimation errors. The αL algorithm always performs better than the N_{MIN} algorithm, but does not succeed in outperforming some of the other algorithms considered. For a given traffic load, its relative performance compared to the other algorithms does not change with the choice of the parameter α (small values of α produce longer bursts as it can be seen in the figures). Among the previous burst assembly schemes (T_{MAX}, L_{MAX}, T_{AVE}), the T_{AVE} algorithm gives the best performance. The proposed T_A algorithm outperforms the T_{AVE} algorithm, but the improvement is

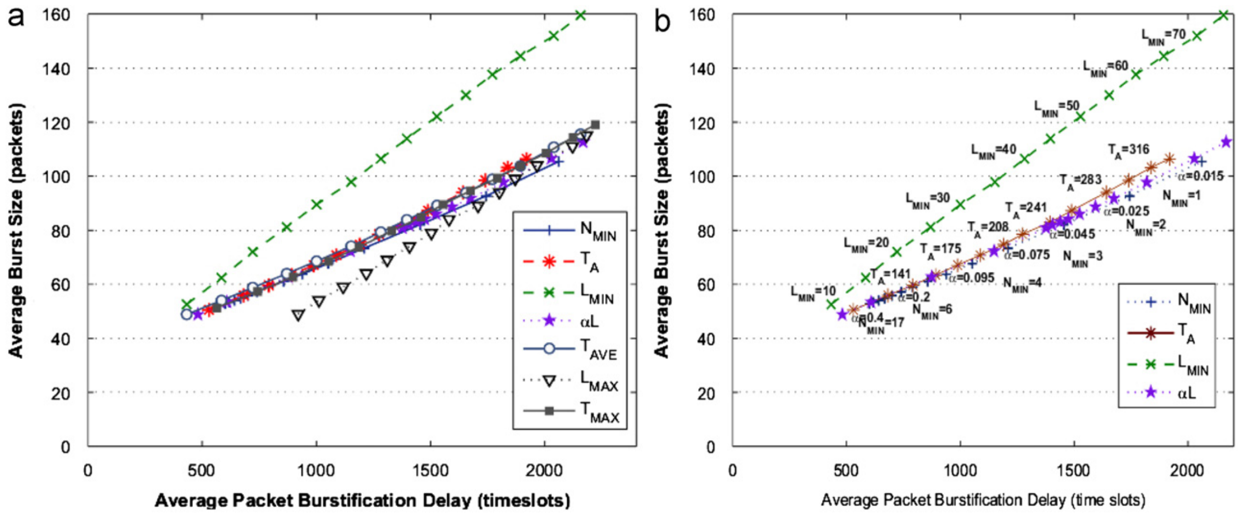

Fig. 8. Performance of the proposed algorithms for traffic load p=0.03: (a) comparison of the proposed schemes with previously proposed algorithms and (b) details on the values of the parameters used in the proposed algorithms.

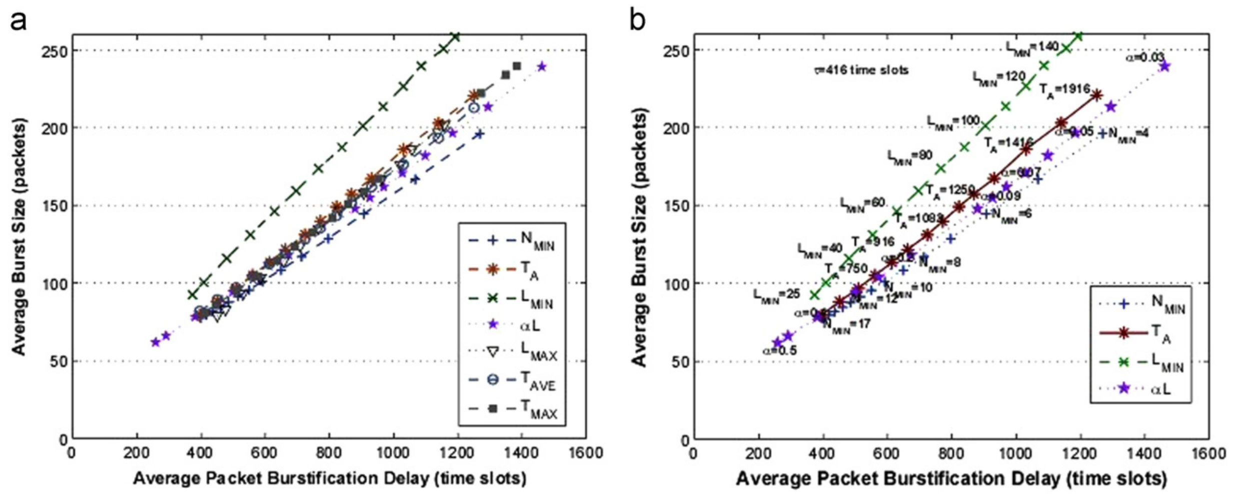

Fig. 9. Performance of the proposed algorithms for traffic load p=0.1: (a) comparison of the proposed schemes with previously proposed algorithms and (b) the parameters applied on the proposed algorithms.

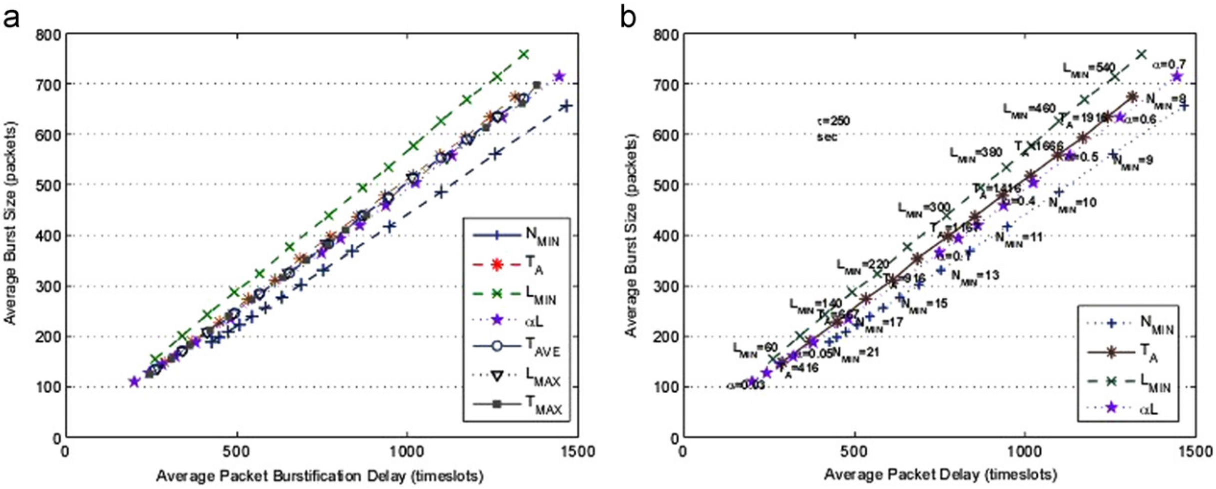

Fig. 10. Performance of the proposed algorithms for traffic load $p=0.3$: (a) comparison of the proposed schemes with previously proposed algorithms and (b) the parameters applied on the proposed algorithms.

rather small, as shown in Fig. 10a. The improvement is more pronounced when the T_A algorithm generates longer bursts and when the traffic load is heavier.

The best performance is consistently demonstrated by the L_{MIN} algorithm, which achieves a 33% improvement over the T_A algorithm (the second best) for light traffic load ($p=0.003$ and 0.1) and an 8% improvement for heavier traffic load ($p=0.3$). For a given average packet burst assembly delay, the L_{MIN} algorithm produces bursts of larger average size than all the other algorithms considered. The L_{MIN} algorithm can be considered a variation of the L_{MAX} algorithm, enhanced with the ability to predict the time periods where the value of DBR is expected to improve because of a large number of future packet arrivals. Note that in most of the figures, the curve that corresponds to the L_{MIN} algorithm is parallel to and above that of the L_{MAX} algorithm.

5.2. *Fast reservation protocols*

In this section we evaluate the performance of the fast reservation (FR) protocol of Section 3, using an OBS network simulator [30] based on the ns-2 platform [31]. The FR protocol was combined with the T_{MAX}, L_{MAX}, and T_{AVE} burst assembly schemes, and used predictions to estimate the corresponding burst length and/or the burst assembly durations.

In the experiments we use a simple OBS network consisting of two edge (ingress and egress) nodes and one core node. A link's bandwidth per channel is equal to 10 Gbps. The arrivals at the ingress node were obtained from a Pareto traffic generating source of rate $r=1$ Gbps, while the mean On and Off periods were equal to 0.002 ms and 0.001 ms respectively. The traffic source generates packets of fixed size, equal to $l=1500$ bytes. In each experiment we change the shape parameter β that determines the Hurst parameter $H=(3-\beta)/2$ and defines the burstiness of the traffic. Specifically, we used the values $\beta=1.2$, 1.4, 1.6, 1.8; the closer the value of H is to 1, the burstier the traffic can be characterized. Also, we use a 16-order LMS filter, even though a filter with a smaller order could also have been used. The value of the step-size μ of the filter was equal to 0.02. For the T_{MAX} assembly algorithm we used the following values for the parameter T_{MAX}: 0.006 s, 0.008 s, 0.01 s. For comparison purposes, the corresponding values for the parameter L_{MAX} of the L_{MAX} assembly algorithm were 488 KB, 651 KB, 813 KB, and were calculated based on

$$L=\rho RD, \tag{25}$$

where D is the average burst assembly duration (T_{MAX} parameter), L is the average burst lengths (L_{MAX} parameter) and R is the rate of the Pareto traffic. The traffic load ρ is defined as

$$\rho=\frac{on_{\text{period}}}{on_{\text{period}}+off_{\text{period}}}. \tag{26}$$

Finally, for the average delay-based assembly algorithm the values of the parameter T_{AVE} we used were one half of the corresponding values used for the parameter T_{MAX} (that is, 0.003 s, 0.004 s, and 0.005 s).

In the experiments we measured the following performance metrics:

- the relative error of the prediction of the burst size and the burst assembly duration, defined as the inverses of the signal-to-noise ratios

  $$SNR^{-1}=\frac{\sum e_L^2(k)}{\sum L^2(k)},$$

  and

  $$SNR^{-1}=\frac{\sum e_D^2(k)}{\sum D^2(k)},$$

  respectively, where $e_L(k)=L(k)-\hat{L}(k)$ and $e_D(k)=D(k)-\hat{D}(k)$
- the probability of the BHP performing a successful reservation. In the T_{MAX} algorithm a reservation is successful if the size of the predicted burst (plus δ) is bigger than the actual burst size, that is, if Eq. (16)

holds. In the L_{MAX} algorithm a reservation is successful if the predicted burst assembly duration (plus ε) is larger than the actual assembly duration, that is, if Eq. (15) holds. Finally in the T_{AVE} algorithm a reservation is considered successful if Eq. (13a) and Eq. (13b) are both valid. Note that a failed reservation does not result in a burst loss, since the estimates are compared to the real values, when the burst assembly is completed, and a new BHP is sent if needed. It only results in some loss of efficiency in the small interval between the reservation and the cancellation, where the resource remains idle.

Generally we want to have small relative errors and a large probability of successful reservations.

We first present the results for the FR protocol when combined with the T_{MAX} and L_{MAX} burst assembly algorithms in order to separately evaluate the two LMS filters used. Subsequently, we present the results obtained for the FR protocol when combined with the T_{AVE} algorithm, where both LMS filters are used.

5.2.1. FR protocol combined with the T_{MAX} algorithm

Fig. 11 illustrates the relative error of the burst length prediction versus the shape parameter β, for different values of the burst assembly duration T_{MAX}. We observe that the relative error of the burst length prediction is quite small and decreases as the shape parameter β increases.

Fig. 12 illustrates the cumulative distribution function (cdf) of the errors of the burst length filter predictions for burst assembly period $D=T_{MAX}=0.01$ s (which corresponds to about 813 KB of burst length) and shape parameter $\beta=1.4$. The average error measured is equal to 0.135831 KB, which is quite small compared to the average 813 KB burst length, and is due to statistical fluctuations (the estimator is unbiased).

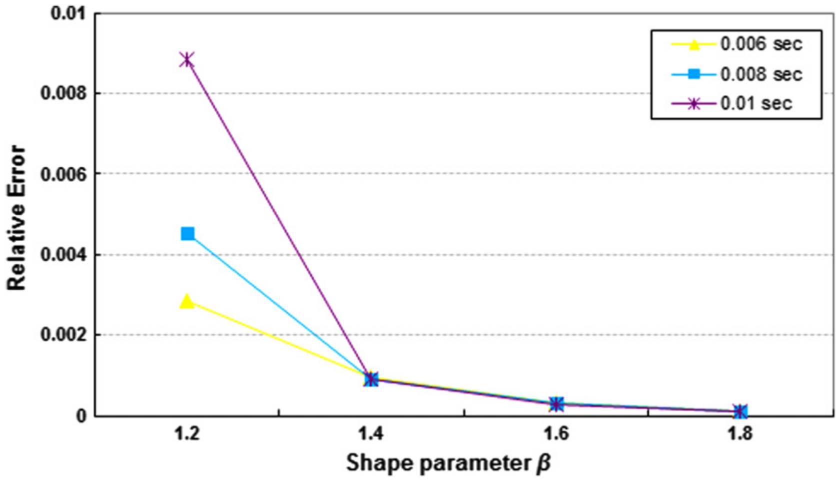

Fig. 11. The relative error of the burst length prediction versus the shape parameter β, for three different values of the burst assembly duration $D=T_{MAX}$ for the T_{MAX} assembly algorithm.

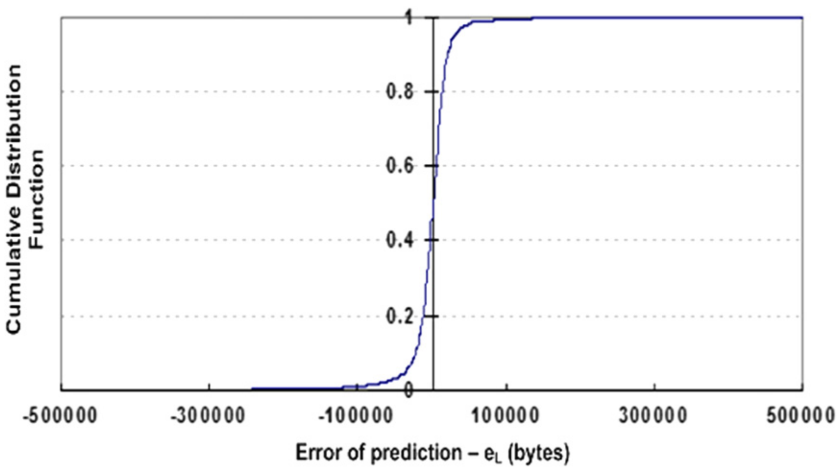

Fig. 12. The empirical cumulative distribution function (cdf) of occurrences of the burst length prediction errors e_L, for the T_{MAX} assembly algorithm and assembly duration $T_{MAX}=0.01$ s.

The probability of successful reservations was also evaluated for safety margins δ that correspond to different correction parameters $c_\delta = 0$, 1, 2, and 3. The experiments were conducted for burst assembly period $D = T_{MAX} = 0.006$ s. Fig. 13 shows the probability of the BHP successfully performing a reservation for various values of the shape parameter β and correction parameter c_δ. We observe that the results obtained for c_δ equal to 2 or 3 are quite satisfactory, and the probability of successful reservations increases as c_δ increases. These results are consistent with the results also presented in [12].

5.2.2. FR protocol combined with the L_{MAX} algorithm

Fig. 14 shows the relative prediction error for various values of the shape parameter β and three different choices for the burst lengths L_{MAX}. These three values correspond to the values we used for the burst assembly duration D, according to Eq. (12). As in the case of the T_{MAX} algorithm, we observe that the relative error of the prediction decreases as the shape parameter β increases.

Fig. 15 illustrates the cumulative distribution function (cdf) of the prediction errors for $L = L_{MAX} = 813$ KB (which corresponds to about 0.01 s burst assembly duration) and shape parameter $\beta = 1.4$. The average prediction error was equal to 0.000194 s, which is quite small (compared to the 0.01 s average assembly duration) and is due to statistical fluctuations. Note that the error empirical cdf is not exactly symmetric (the probability that the error is negative is 0.4), but the error is generally very small.

The probability of successful reservations of the L_{MAX} algorithm was evaluated for different safety margins ε, using correction parameters $c_\varepsilon = 1$, 1.02, 1.05, 1.08, 1.1, 1.5, 1, 2, and burst size $L = L_{MAX} = 813$ KB. Fig. 16 shows the probability of successful reservations for various

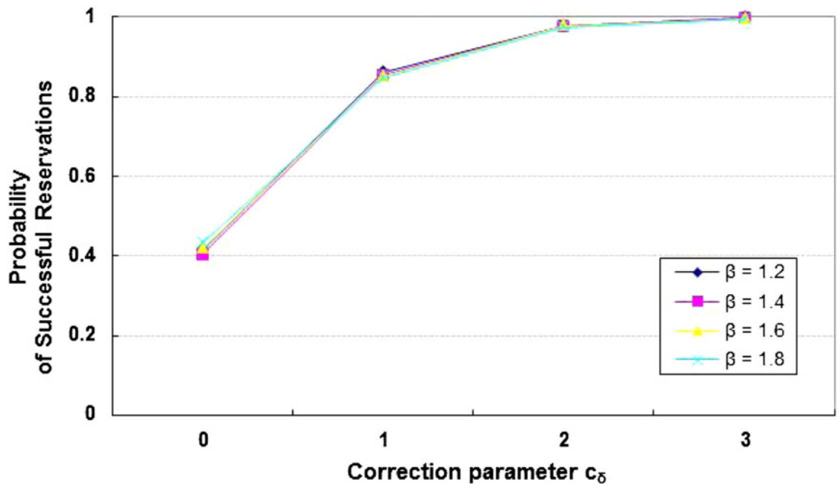

Fig. 13. The probability of successful reservations versus the correction parameter c_δ, for different values of the shape parameter β for the T_{MAX} assembly algorithm. The burst assembly period was $T_{MAX} = 0.006$ s. A choice of 2 or 3 for the correction of the correction parameter c_δ is adequate for obtaining very satisfactory performance.

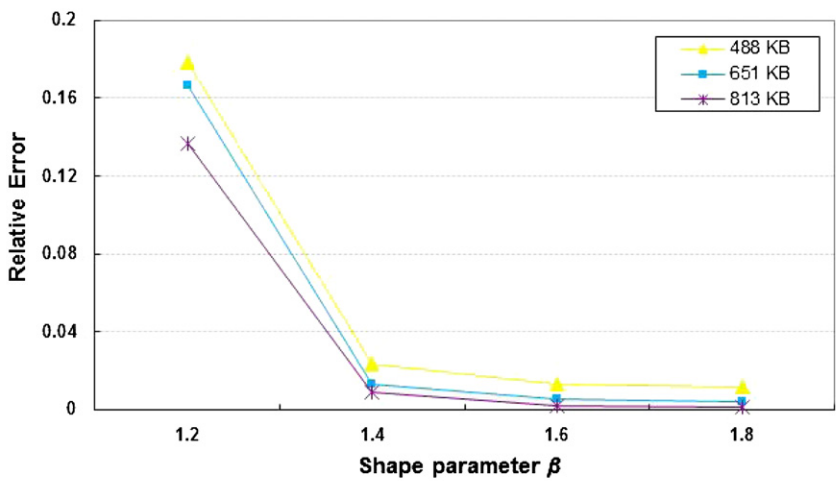

Fig. 14. The relative error of the prediction versus the shape parameter β, for the L_{MAX} assembly algorithm and three different values of the burst size L_{MAX}.

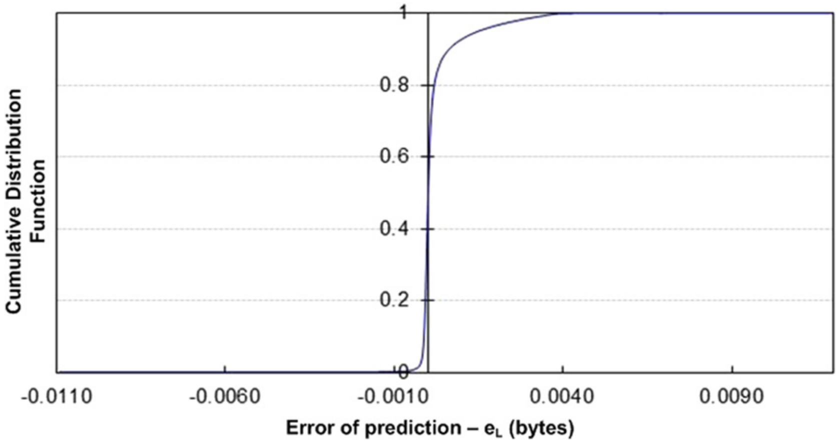

Fig. 15. The cumulative distribution function (cdf) of occurrences of the errors e_D of the burst assembly duration filter predictions, for the L_{MAX} assembly algorithm and burst size $L_{\mathrm{MAX}}=813$ KB.

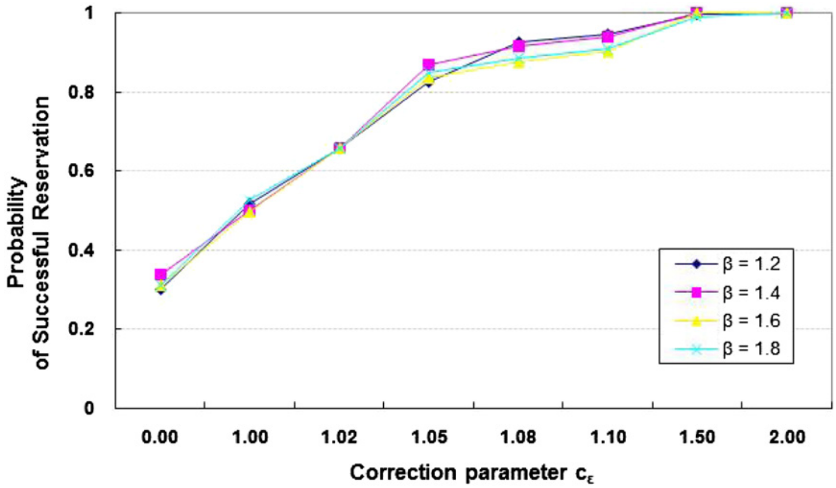

Fig. 16. The probability of successful reservations versus the correction parameter c_ε, for different values of the shape parameter β. The burst size was $L=L_{\mathrm{MAX}}=813$ KB. Choosing c_ε to be larger than 1.5 gives satisfactory performance.

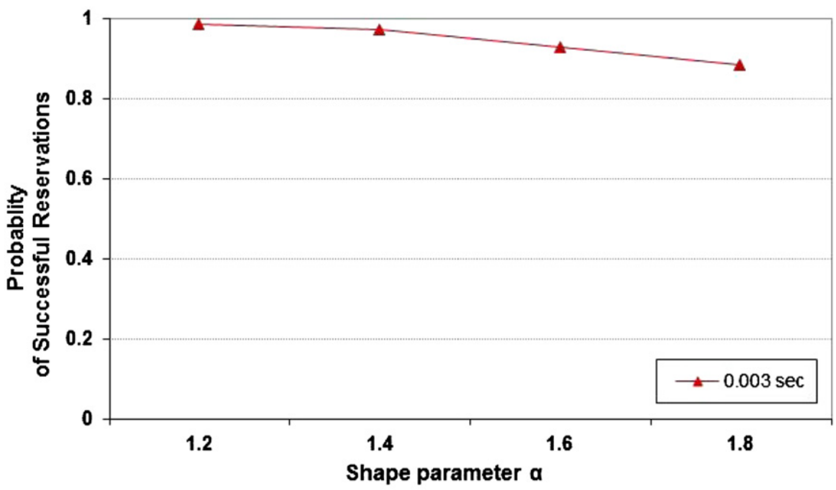

Fig. 17. The probability of a BHP successful reservation versus the shape parameter β, for the T_{AVE} assembly algorithm with $T_{\mathrm{AVE}}=0.003$ s. We used length correction parameter $c_\delta=3$ and assembly duration correction parameter $c_\varepsilon=2$.

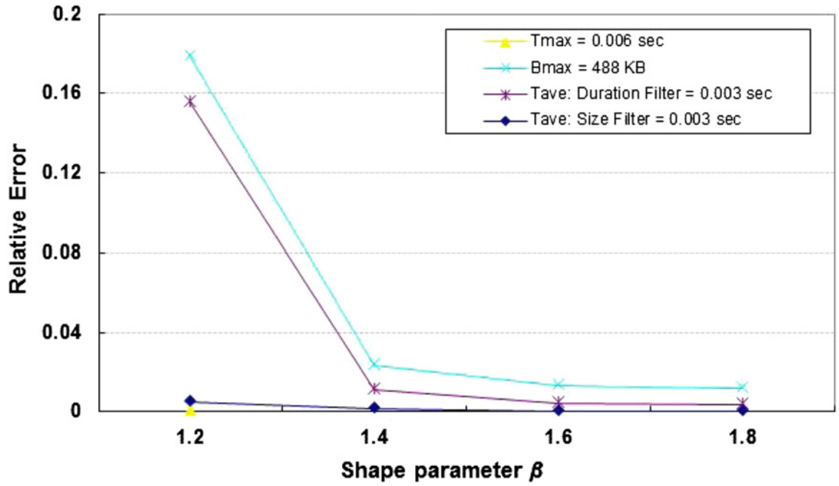

Fig. 18. The relative errors of T_{MAX}, L_{MAX} and T_{AVE} versus the shape parameter β.

values of the shape parameter β and correction parameter c_ε. We observe that choosing $c_\varepsilon=1.5$ or 2 provides sufficiently high success probability and the probability of successful predictions increases as c_ε increases.

5.2.3. *FR protocol combined with the T_{AVE} algorithm*

When the FR protocol is combined with the T_{AVE} assembly algorithm, we have to use two LMS filters to predict the length of the next burst and the burst assembly duration. Fig. 17 shows the probability of successful reservations, using safety margins δ and ε; in particular, we used burst length and assembly duration correction parameter $c_\delta=3$ and $c_\varepsilon=2$, respectively. The success probability is calculated by multiplying the probability of successful predictions of the two LMS filters used. The corresponding experiments were conducted for average delay parameter $T_{AVE}=0.003$ s. From Fig. 17 we see that the success probability is quite high and slightly deteriorates when the shape parameter β of the Pareto traffic increases. As already mentioned a failed reservation does not result in burst loss, but only has some small effect on efficiency.

Fig. 18 compares the T_{MAX}, L_{MAX} and T_{AVE} (for both filters) relative errors, for different values of the Pareto parameter β. Note that the filters predicting burst assembly durations have larger relative errors than those predicting the burst sizes.

6. Conclusions

We proposed four new burst assembly schemes for OBS networks based on traffic prediction. The L_{MIN} assembly scheme seems to be the algorithm of choice when the average burst assembly delay (for a given burst size) or the average burst size (for a given burst assembly delay) is the criterion of interest. One should note, however, that the T_{AVE} and the T_A algorithms may be preferable when the delay jitter is the main consideration (both of these algorithms also give a satisfactory average burst assembly delay to average burst size ratio). We also presented a general formulation for combining fast reservation (FR) schemes with the T_{MAX}, L_{MAX} and T_{AVE} burst assembly algorithms, to further reduce the pre-transmission delay in OBS networks. Moreover, the FR scheme used for the T_{AVE} algorithm can also be combined with the proposed burst assembly schemes (N_{MIN}, αL, T_A, and L_{MIN}). We find that the probability of successful reservations using fast reservations is very satisfactory, provided that a small correction term (2 to 3 times the root mean square error) is added to the predicted burst length and assembly duration estimates. The proposed schemes and protocols can be used to reduce the end-to-end delay and increase the size of the bursts produced, while making efficient use of the bandwidth, and maintaining a good probability of success for the reservations needed in an OBS network.

References

[1] C. Qiao, M. Yoo, Optical burst switching (OBS)—a new paradigm for an optical internet, Journal of High Speed Networks 8 (1999) 69–84.
[2] V. Vokkarane, K. Haridoss, J. Jue, Threshold-based burst assembly policies for QoS Support in optical burst-switched networks, in: Proceedings of the Opticomm, 2003, pp. 125–136.
[3] A. Ge, F. Callegati, L. Tamil, On optical burst switching and self-similar traffic, IEEE Communications Letters 4 (3) (2000).
[4] S. Gowda, R. Shenai, K. Sivalingam, H. Cankaya, Performance evaluation of TCP over optical burst-switched (OBS)wdm networks, in: Proceedings of the Communications, ICC '03, IEEE International Conference, IEEE Computer Society Press, Los Alamitos May 2003, pp. 11–15.
[5] X. Yu, Y. Chen, C. Qiao, Study of traffic statistics of assembled burst traffic in optical burst switched networks, in: Proceedings of the Opticomm, pp. 149–159, 2002.
[6] X. Cao, J. Li, Y. Chen, C. Qiao, Assembling TCP/IP packets in optical burst switched networks, IEEE Globecom, 2002.
[7] K. Christodoulopoulos, E. Varvarigos, K. Vlachos, A new burst assembly scheme based on the average packet delay and its performance for TCP Traffic, Optical Switching and Networking 4 (3-4) (2007) 200–217.
[8] K. Ramantas, K. Vlachos, Ó.G. de Dios, C. Raffaelli, Window-based burst assembly scheme for TCP traffic over OBS, Journal of Optical Networking 7 (5) (2008) 487–495.
[9] S. Ozsarac, E. Karasan, Congestion window-based adaptive burst assembly for TCP traffic in OBS networks, Photonic Network Communications (2010) 138–150.

[10] S. Ostring, H. Sirisena, The Influence of Long-range Dependence on Traffic Prediction, in: Proceedings of ICC, 2001.

[11] D. Morato, J. Aracil, L. Diez, M. Izal, E. Magana, On linear prediction of Internet traffic for packet and burst switching networks, in: Proceedings of the 10th International Conference on Computer Communications Networks, 2001, pp. 138–143.

[12] J. Liu, N. Ansari, T. Ott, FRR for latency reduction and QoS provisioning in OBS networks, IEEE Journal on Selected Areas in Communications 21 (7) (2003).

[13] K. Ramantas, K. Vlachos, A TCP prediction scheme for enhancing performance in OBS networks, in: Proceedings of the IEEE International Conference on Communications, 2011.

[14] E. Varvarigos, V. Sharma, An efficient reservation connection control protocol for gigabit networks, Computer Networks and ISDN Systems 30 (12) (1998) 1135–1156. July.

[15] M. Dueser, P. Bayvel, Analysis of a dynamically wavelength-routed optical burst switched network architecture, Journal of Lightwave Technology 20 (2002) 574–585.

[16] K. Christodoulopoulos, K. Vlachos, K. Yiannopoulos, E. Varvarigos, Relaxing delayed reservations: an approach for quality of service differentiation in optical burst switching networks, Broadnets, 2006.

[17] E. Varvarigos, V. Sharma, The ready-to-go virtual circuit protocol: a loss-free protocol for multigigabit networks using FIFO buffers, IEEE/ACM Transactions on Networking 5 (5) (1997) 705–718.

[18] J. Turner, Terabit burst switching, Journal of High Speed Networks 8 (1999) 3–16.

[19] M. Yoo, C. Qiao, Just-enough-time (JET): a high speed protocol for bursty traffic in optical networks, in: Proceedings of IEEE/LEOS Conference on Technologies for a Global Information Infrastructure, August 1997, pp. 26–27.

[20] M. Yoo, C. Qiao, S. Dixit, Optical burst switching for service differentiation in the next-generation optical Internet, IEEE Communication Magazine 39 (2001) 98–104.

[21] J.Y. Wei, R.I. McFarland., Just-in-time signaling for WDM optical burst switching networks, Journal of Lightwave Technology 18 (12) (2000) 2019–2037.

[22] I. Baldine, G.N. Rouskas, H.G. Perros, D. Stevenson., JumpStart: a just-in-time signaling architecture for WDM burst-switched networks, IEEE Communications 40 (2) (2002) 82–89.

[23] J. Teng, G.N. Rouskas, A detailed analysis and performance comparison of wavelength reservation schemes for optical burst switched networks, Photonic Network Communications 9 (3) (2005) 311–335.

[24] J.J.P.C. Rodrigues, J.M.B. Gregório, A.V. Vasilakos, Enhanced just-in-time plus protocol for optical burst switching networks, Optical Engineering 49 (7) (2010).

[25] N.M. Garcia, P. Lenkiewicz, M.M. Freire, P.P. Monteiro, A new architecture for optical burst switching networks based on cooperative control, IEEE International Symposium on Network Computing and Applications (NCA) (2009).

[26] K. Seklou, E. Varvarigos, Fast reservation protocols for latency reduction in optical burst-switched, networks based on predictions, International Conference on Networking (ICN), 2008, pp 656–662.

[27] M.F. Zhani, H. Elbiaze, F. Kamoun, Analysis and prediction of real network traffic, Journal of Networks 4 (9) (2009) 855–865.

[28] Sang Aimin, Li San-qi, A predictability analysis of network traffic, Computer Networks 39 (4) (2002) 329–345.

[29] J. Treicher, C. Johnson Jr., M. Larimore, Theory and Design of Adaptive Filters, Wiley, New York, 1987.

[30] Optical burst switching simulator based on ns-2. Available from: 〈http://wine.icu.ac.kr/~obsns/index.php〉.

[31] The Network Simulator on ns-2. Available from: 〈http://www.isi.edu/nsnam/ns/index.html〉.