IET Networks

Research Article

IET Journals
The Institution of Engineering and Technology

Burst-by-burst dynamic bandwidth allocation for XG-PONs

ISSN 2047-4954
Received on 10th July 2015
Revised on 1st November 2015
Accepted on 23rd November 2015
doi: 10.1049/iet-net.2015.0070
www.ietdl.org

Ilias Gravalos[1,2] ✉, *Kostas Yiannopoulos*[1,3], *Georgios Papadimitriou*[4], *Emmanuel A. Varvarigos*[1,2]

[1]*Computer Technology Institute and Press 'Diophantus', Rio 26500, Greece*
[2]*Computer Engineering and Informatics Department, University of Patras, Rio 26500, Greece*
[3]*Department of Informatics and Telecommunications, University of Peloponnese, Tripoli 22100, Greece*
[4]*Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki 54124, Greece*
✉ *E-mail: grabalos@ceid.upatras.gr*

Abstract: The authors propose and evaluate the performance of a dynamic bandwidth allocation algorithm, namely burst-by-bUrst dynamic bandwidth allocation (BUDA), which enables the fair and efficient utilisation of the available upstream resources in 10-Gigabit-capable passive optical networks (XG-PONs). The proposed algorithm emulates a bit-by-bit burst multiplexing process based on the fair queuing scheduling principle to fairly allocate the available upstream bandwidth among several end-users with varying traffic demands. The fair queuing approach is well-suited to the strictly synchronous communication model of XG-PONs, where fixed duration downstream and upstream frames are utilised to delineate transmissions, and the authors demonstrate via simulations that BUDA achieves low and bounded latency, as well as resource allocation fairness over a set of different traffic loads. Furthermore, BUDA is compared with well-known bandwidth allocation algorithms and a significant latency and fairness benefit is demonstrated.

1 Introduction

As the bandwidth demands ever increase in the network access segment, optical networks keep up to challenges and provide ever-increasing bandwidth to the end-user. When cost considerations also come into play, passive optical networks (PONs) constitute a cost-effective solution for the fibre-to-the-x access models that are envisaged as the next logical step towards truly broadband access. This owes to the fact that PONs can fully exploit the passive nature of the optical infrastructure and provide high capacities, enabling the delivering integrated video, voice and data services [1–3]. PONs are implemented over a point-to-multipoint architecture that consists of a central optical line terminal (OLT) located at the service provider end, and multiple optical network units (ONUs) that are connected through a passive optical distribution network in a distributed tree-like topology. In XG-PONs, the OLT communicates with the ONUs by broadcasting downstream frames that are transported over a single wavelength. On the reverse direction, an upstream frame that operates on a different wavelength is distributed among the ONUs and each ONU periodically sends its data in the form of optical bursts. Due to the multiple access of ONUs in the upstream direction, a collision avoidance mechanism is necessary for reliable data transmission and delivery at the OLT. Since channel sensing is not an option in the tree-like PON architecture, the multiple access functionality is implemented by polling and the OLT is responsible for resource allocation and burst scheduling in the PON, as well as for orchestrating the ONU transmissions in a manner that avoids upstream collisions.

Following the above, the OLT must feature a resource allocation mechanism to distribute the available upstream bandwidth on a per frame basis. A dynamic bandwidth allocation (DBA) algorithm is also preferable to ensure that varying demands will be served, while in the dynamic operation the bandwidth arbitration can be adapted to the instantaneous traffic demands of each ONU and therefore bandwidth utilisation is improved. To this end, we propose a novel DBA algorithm named burst-by-bUrst dynamic bandwidth Allocation (BUDA) that provides fairness, low latency and full utilisation of the upstream bandwidth. BUDA utilises the FQ policy to allocate the available upstream bandwidth in a bit-by-bit manner [4], based on the traffic demands that the OLT has acquired from the ONUs. Whenever traffic requests exceed the upstream frame capacity, the requests that cannot be fully served are partially deferred for the next frame and are served in a bit-by-bit manner but with priority over subsequent requests. In addition, if traffic requests are not sufficient to fill up the upstream frame, which is typically expected in low network loads, a rate-proportional excess bandwidth redistribution scheme is enforced to overprovision ONUs and enable the rapid transmission of data without reporting. The assessment of the algorithm, which was performed via simulation, shows that the algorithm exhibits low latency even at high network loads, while at the same time it provides identical capacity to all ONUs without incurring bandwidth losses. Finally, BUDA is shown to significantly outperform well-known DBAs in terms of latency and fairness.

The remainder of this paper is organised as follows. In Section 2 we provide a background review on PON DBA algorithms and works related to the FQ scheduling principle. In Section 3 we perform a brief overview on the XG-PON recommendation, giving emphasis on the synchronisation and communication procedures, structures and fundamental operations that are relevant to this work. In Section 4 we present the main idea of the proposed DBA algorithm, as well as a description of compatible scheduling options. Finally, in Section 5 we evaluate via simulation the throughput, latency and fairness performance of BUDA and compare it with the well-known DBAs that have been previously proposed for E-PONs and XG-PONs.

2 Background

DBA is a crucial functionality in both XG-PONs and E-PONs, and has received considerable attention in works that aim to efficiency utilise the shared upstream bandwidth. In E-PONs, which have been standardised by IEEE within the 802.av framework [5], bandwidth allocation was initially investigated via interleaved polling with adaptive cycle time (IPACT) scheduling algorithms that provided increasing levels of dynamicity [6], ranging from a

limited service where ONUs only receive resources up to a predefined maximum, to a *gated* service where ONUs always receive the requested resources [7]. IPACT variants exhibit a different delay performance, with gated service outperforming the rest, and this observation instigated an extensive study of bandwidth allocation schemes that could improve the delay aspects in E-PONs [8, 9]. These include excess resource allocation schemes, where any unutilised bandwidth is systematically assigned to overloaded ONUs [10, 11], and traffic prediction techniques [12], where an estimate of (rather than the reported) ONU request is processed in the DBA. Service differentiation was also studied as an extension to IPACT, and DBAs that take account the intra-ONU classification of data packets and implement scheduling in a flat [13] or hierarchical [14] model have been presented.

XG-PONs, which are standardised by ITU [15], have not received equally significant attention and only a few DBA algorithms have been proposed exclusively for XG-PONs [16–21]. XG-PONs are engineered for the transport of time-critical data and the ITU-T recommendation enforces stringent timing requirements on their operation [22], thus inherently limiting the data latency. Moreover, service differentiation is also addressed by the ITU standard via 'traffic containers' that are treated separately by the DBA depending on their priority. As a result, previous works on XG-PON specific DBAs focus on satisfying service-level-agreements (SLAs) and allocate resources to ONUs over predefined service intervals (SI) based on down-counters [16]. This approach leads to limitations on the peak rate that can be assigned to each ONU and, as a result, several modifications of the down-counter model have been proposed to improve the PON latency and frame loss [17], and adapt the original G-PON DBA to the XG-PON environment [18, 19].

Despite the fact that the existing DBAs provide a pre-determined degree of SLA, their main drawback is that they are not closely adapting to the instantaneous traffic demands of ONUs and bandwidth waste can be observed, especially for best-effort traffic. To address this issue, we recently proposed a max–min fair DBA that is suitable for XG-PONs and distributes resources among ONUs in an increasing order of demand [21]. Our previous work, however, did not take into account the start and finish times (FT) of ONU bursts, and if an ONU was partially served then its remaining request had to compete for bandwidth with newly arrived requests in the upcoming frame. In this work, we take into account the timing components of requests (arrival and FT) in a manner that is dictated by the FQ principle, with a goal to further reduce latency in the XG-PON. The FQ principle ensures that all requests will be served in a fair and efficient manner by emulating bit-by-bit service [23], and a number of FQ variants have been previously reported [24–27], as well as numerous applications in wireless and wired networks [28–30]. To our knowledge, the application of FQ in PONs has been previously reported only in terms of providing fairness among ONUs in a hierarchical scheduling topology [14].

3 XG-PON and E-PON communication procedures

In this section we present key definitions and standardised communication procedures that are defined in the G.987.3 ITU-T recommendation [15], so as to better explain the operation of the proposed algorithm. Similar to E-PONs, the XG-PON recommendation defines two data transmission paths:

(i) the 'downstream' (D/S) path that operates at a nominal wavelength of 1577 nm and is used for transmitting the data from the OLT to the ONUs at a data rate of 9.95328 Gb/s, and
(ii) the 'upstream' (U/S) path that operates at a nominal wavelength of 1270 nm and is used for transmitting the data from the ONUs to the OLT at a data rate of 2.48832 Gb/s.

In contrast with E-PONs that are designed to transport Ethernet traffic, XG-PONs are also required to support real-time applications, including telephony, which mandates a constant flow of data every 125 µs. Consequently, both the downstream and the upstream directions in the XG-PON operate over 125 µs long timeslots, and all communications take place within the timeslot boundaries, as it is shown in Fig. 1*a*. In the downstream direction the timeslot fits exactly one downstream frame and this frame transports data for all ONUs in its payload. The size of the downstream frame corresponds to 155,520 bytes (38,880 words). In the upstream direction XG-PONs follows a similar timeslot convention, but the upstream frame comprises several 'bursts' of data that are sent by the ONUs. The upstream frame size equals 38,880 bytes (9720 words) and bursts are separated by a small guard-band (usually 64 bits).

Given that the ONU bursts are required to arrive within a common timeslot in XG-PONs, a synchronisation process is required:

(i) The OLT first estimates the maximum delay from the beginning of a downstream frame to the beginning of the upstream frame (T_{eqd}). This delay accommodates the round-trip delay to the most distant ONU, and also takes into account the maximum response time of ONUs. The response time corresponds to the time that an ONU hardware takes to process the downstream frame and prepare an upstream burst (typically 35 ± 1 µs).
(ii) The OLT then determines an 'equalisation delay' (*EqD*) for each ONU. The equalisation delay must always be observed by ONUs after the reception of the downstream frame and before the transmission of the upstream burst.

E-PONs, on the other hand, operate in an asynchronous manner shown in Fig. 1*b*. The OLT serially sends downstream bursts to ONUs, and also schedules ONU bursts in the upstream by only taking into account the round-trip delay of each respective ONU.

The requirement for synchronisation in XG-PONs introduces a second key difference with E-PONs, which relates to the reporting process. In E-PONs, the OLT sends multiple ONU-specific *GATE* messages that carry bandwidth assignments and start times to ONUs. Each ONU delays its transmission until the corresponding start time has elapsed, so as to ensure that its burst will not collide with another burst, and then transmits the bytes it has been allocated by the OLT. Upstream bursts also contain *REPORT* messages that convey the buffer occupancy of the ONU to the OLT. In XG-PONs, the single downstream frame carries bandwidth assignments and start times for all ONUs in its header, and specifically in the *GrantSize* and *StartTime* fields of the 'bandwidth map' (*BWmap*) structure, as it is shown in Fig. 2*a*. ONUs are then required to delay their transmission by the start time in addition to the equalisation delay, so that bursts arrive to the OLT within the same upstream frame and without suffering from collisions. Upstream bursts in the XG-PON carry the buffer occupancy of the ONUs in the *DBRu* field of their header (refer to Fig. 2*b*).

4 BUDA algorithm

Following the above, the DBA algorithm in both the E-PONs and the XG-PONs is required to efficiently allocate the available upstream bandwidth and determine the start time of each ONU so as to avoid collisions. A DBA algorithm should also take into account fairness so that ONUs are not prioritised over others and within this context we propose the BUDA algorithm that aims to estimate *GrantSize* and *StartTime* values based on the FQ scheduling policy. By strict definition the fair queuing algorithm serves packets that are locally stored at buffers in bit-by-bit round robin fashion. Since bit-by-bit service is not fully feasible on real networks, where packets arrive at random times and have random sizes, generalised processor sharing (GPS) was proposed as a viable alternative [23]. In the GPS approach, the server has knowledge of the buffer occupancies and performs a virtual FQ algorithm to estimate packet FT and thus the service order of the packets.

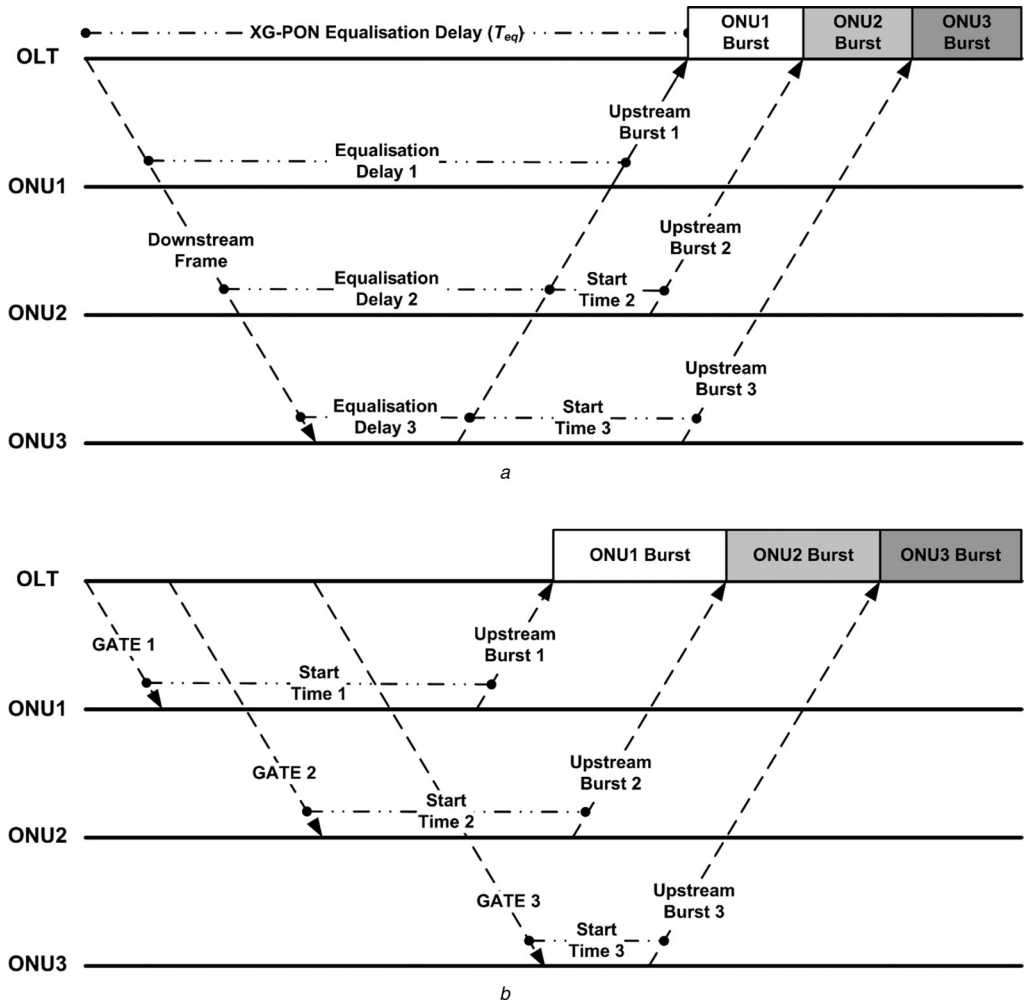

Fig. 1 *Communication procedures in passive optical networks*
a XG-PON
b E-PON

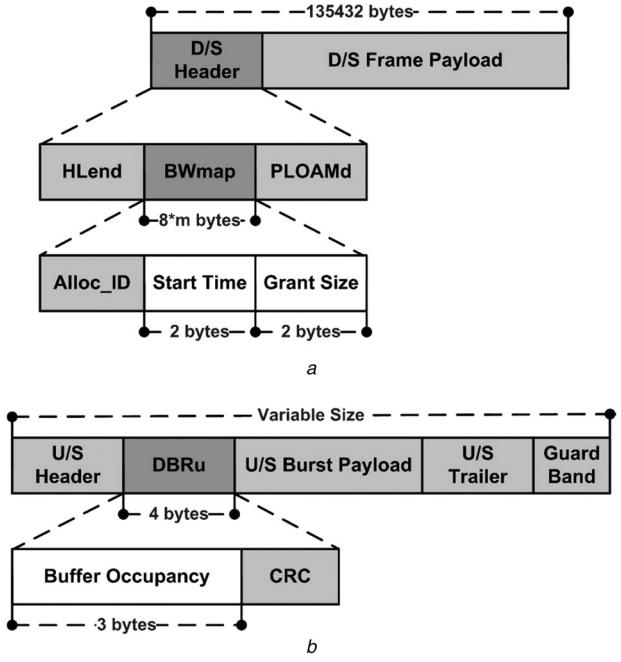

Fig. 2 *Structure of XG-PON frames and bursts*
a Downstream frame
b Upstream burst

The implementation of a FQ algorithm can be challenging in PONs due to their distributed nature, since the server (OLT) is unable to continuously monitor the packet arrivals at the ONU buffers and therefore their occupancies. Following a similar rationale with GPS, however, it is possible to apply the FQ principle on the reported bursts, rather than individual packets, and derive the corresponding burst FT. This is a less accurate approach to FQ than GPS, and the success of this method will depend on how often the ONUs are able to report the buffer occupancies. If a frequent reporting scheme is enforced, for example multi-thread polling [31], then the OLT has a more accurate view of the state of the ONU buffers and the burst FQ algorithm will give more accurate results. Still, status reporting in E-PONs and XG-PONs is performed at a much slower pace and once per service cycle. Given that the E-PON cycle may last several ms [6], the FQ-based burst scheduling principle may prove inaccurate; XG-PONs, on the other hand, require strict synchronisation between frames (125 μs in the same direction or T_{eqd} between directions), and the occupancy reports are also synchronised in the frame level. As a result, a FQ-based DBA algorithm is better suited for XG-PONs due to the smaller time-scales involved.

In BUDA, the XG-PON OLT requires from the ONUs to constantly report their buffer occupancies and then emulates a bit-by-bit serving process over the reported bursts to estimate the total *GrantSize* that each ONU will be allocated given the upstream frame size limit. BUDA then determines the *StartTime* for each burst to ensure that upstream transmissions do not collide. Moreover, due to the constrained upstream frame bandwidth, it is

Table 1 Scheduling algorithm parameters

Parameter	Description
BW	total upstream available bandwidth
$N(\delta)$	set of virtual serving start or finish events
BW_j^p	the available bandwidth for each burst when there are $\left\|\delta_j^n\right\|$ simultaneous transmissions
R_t^n	occupancy demand of ONU n at frame t
W_t^n	assigned burst length (*GrantSize*) of ONU n at frame t
$W_t'^n$	assigned burst length of ONU n at frame t after overprovisioning
m	total number of ONUs
$BW_{residual}$	unutilised bandwidth for upstream frame t
ST_t^n	transmission starting word (*StartTime*) of ONU n at frame t

expected that many of the ONUs' requests will only be partially served. In the presented approach, the portion of the request that is left unserved will explicitly participate in the next frame bandwidth allocation process. Two algorithm variations for the alignment of request remainders in the upcoming frame are also considered:

(i) In the first variation, the request remainders are getting virtually served at the beginning of the next upstream frame along with new requests that arrived to the OLT (FQ-Align). Intuitively, in this approach, all occupancy reports are managed as they had simultaneously arrived at the OLT in the corresponding frame.

(ii) In the second variation, the request remainders are being served first, taking priority over new requests (FQ-Spatial). Each new request then receives a virtual start time that either equals the finish time of the previous remainder from the same ONU, or all new requests are aligned to the finish time of the larger remaining request.

In both variations request remainders are subtracted from ensuing occupancy reports so as to avoid duplicating the requested bandwidth, since remainders are already considered for the next frame in BUDA.

Irrespective of the abovementioned variations in the way requests and their remainders are treated with respect to their virtual start time, the *GrantSize* estimation process of the BUDA algorithm is as follows (Table 1 summarises the definition of the algorithm parameters):

- *Initialisation:* Create an event vector $\boldsymbol{E}$ that stores all the burst request events sorted by the estimated virtual serving start time as explained above. An event specifies the beginning of serving a requested demand or the end of one.
- *Step 1:* For event δ, define the current set of $N(\delta)$ of requests that are being served simultaneously as follows:

$$N(\delta) = \begin{cases} N(\delta-1) \cup \{n_k\} & \text{if burst request } n_k \text{ starts its service period} \\ N(\delta-1) - \{n_k\} & \text{if burst request } n_k \text{ fully served service.} \end{cases} \quad (1)$$

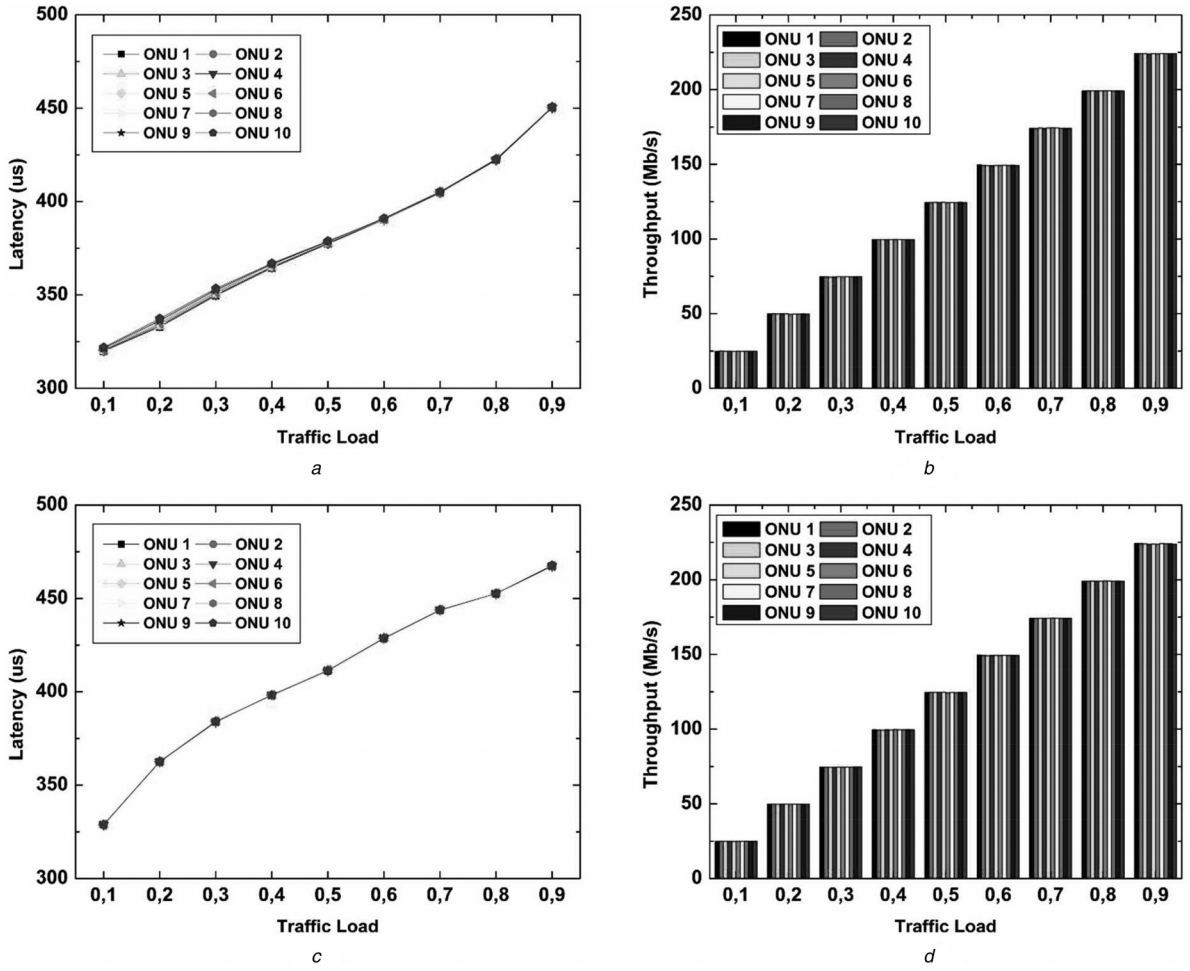

Fig. 3 *Average latency and throughput of BUDA FQ-Spatial for finish time and rotation order scheduling*

a BUDA FQ-spatial (FT) latency (10 ONUs)
b BUDA FQ-spatial (FT) throughput (10 ONUs)
c BUDA FQ-spatial (rotation) latency (10 ONUs)
d BUDA FQ-spatial (rotation) throughput (10 ONUs)

• *Step 2:* For each burst request n_i of the current event estimate respective new FT $F(n_i)$ as follows

$$F(n_i) = \sum_{j=1}^{|\delta^n|} \frac{R_j^n}{\mathrm{BW}_j^p},$$
$$R_j^n = R^n - \mathrm{BW}_j^p \tau\left(\delta_j - \delta_{j-1}\right), \quad (2)$$
$$\mathrm{BW}_j^p = \frac{\mathrm{BW}}{\left|\delta_j^n\right|},$$

where $\tau(\delta_j - \delta_{j-1})$ is the time period between consecutive events, and δ^n are the individual start or finish transmission events of the corresponding burst requests that happens while burst request n_i is being served. R_j^n equals the remaining data after $j-1$ events and BW_j^p is the available bandwidth for each burst when there are $\left|\delta_j^n\right|$ simultaneous transmissions.

• *Step 3:* Update the vector $\boldsymbol{E}$ with the new FT of events as estimated in Step 2. Repeat the process from Step 1 until the iteration of vector $\boldsymbol{E}$ reaches the end.

• *Step 4:* When the final FT have been estimated, the vector $\boldsymbol{E}$ contains all valid start and finish events of the current bursts set. According to the events set, estimate for each ONU the corresponding valid *GrantSize* W_t^n, with respect to upstream frame limits, as follows

$$W_t^n = \sum_{j=1}^{\tau(U_t)} \mathrm{BW}_j^p \tau\left(\delta_j - \delta_{j-1}\right), \quad (3)$$

where $\tau(U_t)$ is the upstream frame finish time.

• *Step 5:* Calculate any remainder burst requests to be accounted for the next upstream frame and return to initialisation.

Finally, BUDA also incorporates a bandwidth overprovisioning scheme for maximum resource utilisation whenever the ONU requests are not adequate to fully occupy an upstream frame. The unused (residual) bandwidth $\mathrm{BW}_{\mathrm{residual}}$, which would be otherwise wasted, is distributed to ONUs in a rate proportional (RP) fashion, assuming that ONUs with a temporarily high buffer occupancy will continue to amass data faster than the others,

$$W_t'^n = W_t^n + \frac{R_t^n}{\sum_{n=1}^m R_t^n} * \mathrm{BW}_{\mathrm{residual}}$$
$$= W_t^n + \frac{R_t^n}{\sum_{n=1}^m R_t^n} * \left(\mathrm{BW} - \sum_{k=1}^m W_t^k\right), \quad (4)$$

with BW denoting the total upstream bandwidth. This technique enables the rapid transmission of data that have arrived to the ONUs without actually ever reporting them, which would impose

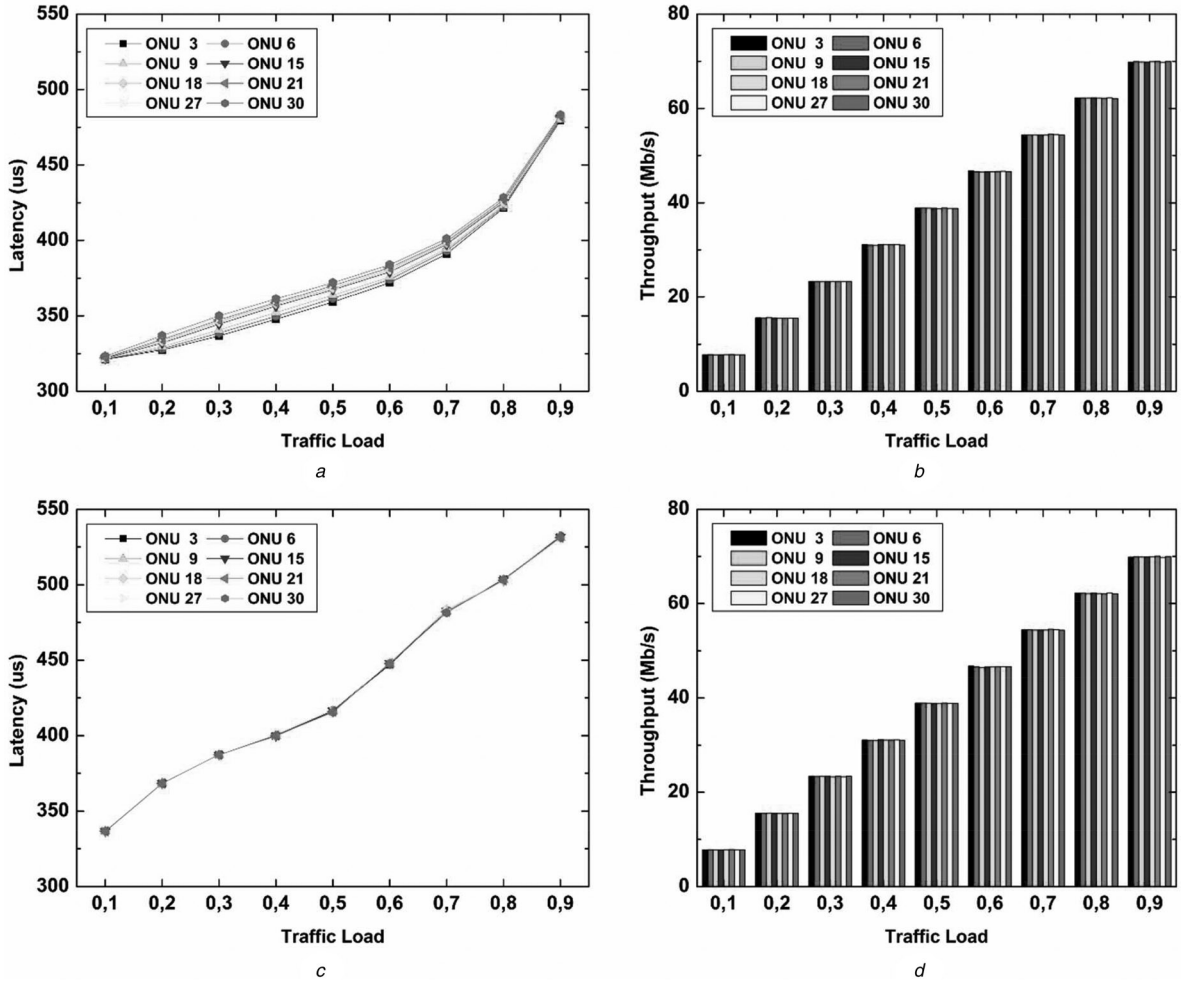

Fig. 4 *Average latency and throughput of BUDA FQ-Spatial for finish time and rotation order scheduling*

a BUDA FQ-spatial (FT) latency (32 ONUs)
b BUDA FQ-spatial (FT) throughput (32 ONUs)
c BUDA FQ-spatial (rotation) latency (32 ONUs)
d BUDA FQ-spatial (rotation) throughput (32 ONUs)

a delay of an additional T_{eqd}, and therefore reduce the frame latency under low loads. With respect to fairness, the RP bandwidth redistribution does not practically affect the DBA operation since it is only invoked when resources are about to be wasted.

After the *GrantSize* has been estimated for all ONUs, the OLT has to determine the *StartTime* values for each transmission in order to orchestrate upstream transmission and avoid collisions. Two approaches were considered: in the first approach, ONU transmissions are scheduled based on the virtual FT of the corresponding requests, and ONUs with earlier FT transmit first. This approach slightly prioritises the closest ONU(s) when two or more ONUs report the same amount of bytes in a frame, and a rotation in the scheduling order of equal-load ONUs was also investigated. In both approaches the *StartTime* ST_t^n of an ONU n during the upstream frame t is calculated as

$$\mathrm{ST}_t^n = \mathrm{ST}_t^{n-1} + W_t'^{n-1} + GuardBand \tag{5}$$

5 Performance evaluation

To assess the performance of BUDA, all related XG-PON subsystems (OLT, ONU, optical splitter, traffic generators) were implemented in the OMNET++ simulation suite [32]. The simulation schematics followed the conventions of the XG-PON standard in terms of line rates, synchronisation and communication procedures, while two topologies with 10 or 32 ONUs randomly placed at distances between 1 and 20 km were considered. In addition, each ONU was connected to a traffic generator that generated variable size packets following a bi-modal distribution [33], where (i) 40% of the packets corresponded to TCP acknowledgements and had a size of 40 bytes, (ii) 40% of the packets had a size equal to the 1500 bytes (maximum Ethernet frame size), and (iii) the rest 20% of the packets had a size that was uniformly distributed between 40 and 1500 bytes. The packet inter-arrival times followed a Poisson distribution with mean value λ that was estimated by the desired ONU load ρ_{ONU} as

$$\rho_{\mathrm{ONU}} = \frac{\rho_{\mathrm{PON}}}{m} = \frac{\lambda \cdot L_{\mathrm{av}}}{m \cdot R_{\mathrm{u}/s}}, \tag{6}$$

where ρ_{PON} is the total network load, m equals the number of ONUs, $R_{\mathrm{u}/s}$ is the upstream data rate and L_{av} is the average frame size in the bi-modal distribution (770 bytes). In the simulation experiments, the average packet latency and the throughput that is achieved in BUDA was measured. The average latency corresponded to the time that a packet spent on average in the ONU buffer from the moment it arrived and until it was transmitted in an upstream burst, while throughput was calculated from the average number of frames (bytes) that are transmitted at simulation over the total simulation time.

Figs. 3 and 4 depict average delay and throughput measurements that BUDA achieves among ONUs when the FQ-Spatial principle is enforced for both the *FT* and *rotation* scheduling principle, respectively. Similar results have been obtained for the FQ-Align principle, but they bear close resemblance and are not shown for

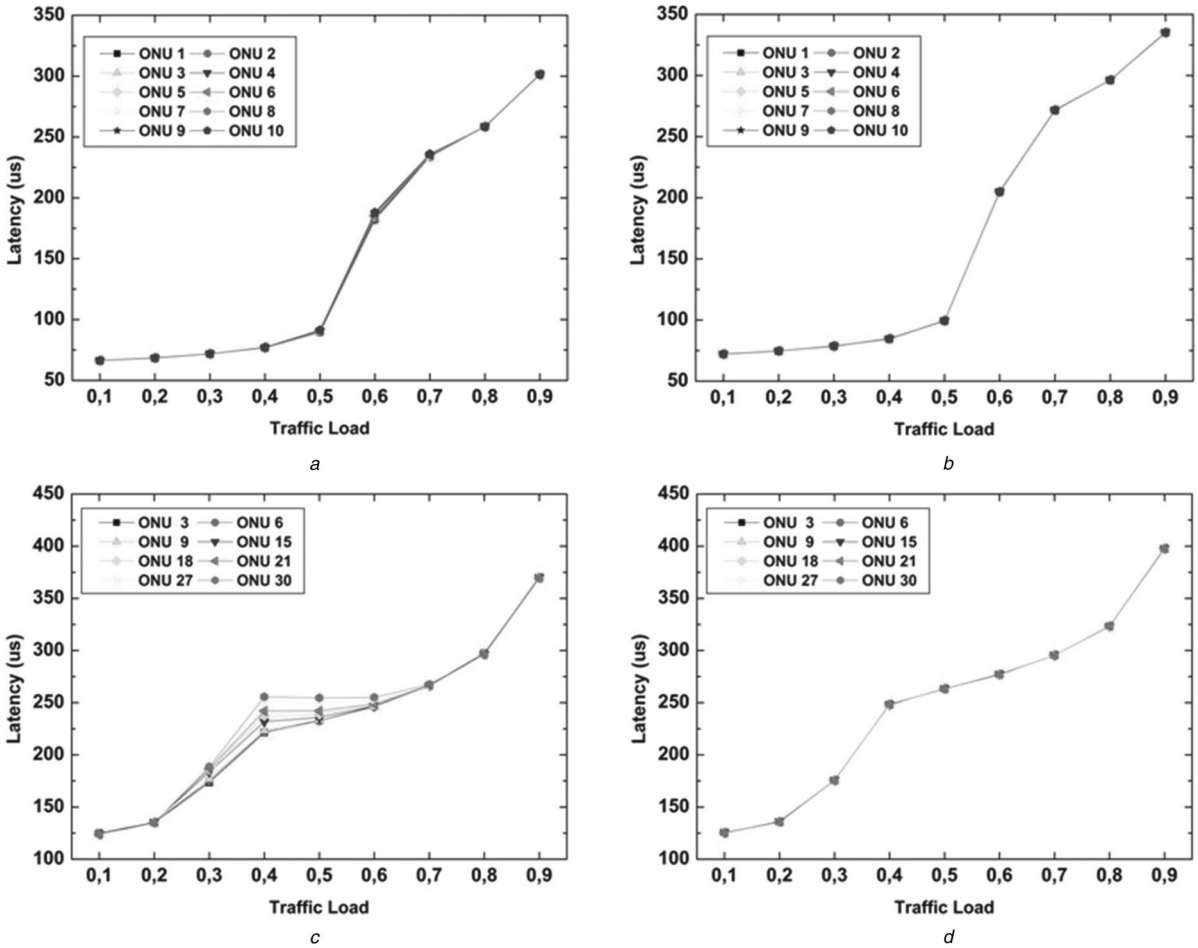

Fig. 5 *Average latency of BUDA FQ-Spatial for finish time and rotation order scheduling with RP*

a BUDA FQ-spatial (FT) latency (10 ONUs)
b BUDA FQ-spatial (rotation) latency (10 ONUs)
c BUDA FQ-spatial (FT) latency (32 ONUs)
d BUDA FQ-spatial (rotation) latency (32 ONUs)

brevity. It becomes evident from these figures that BUDA achieves a latency of less than five frames for both topologies, even at high network loads. Moreover, almost identical latency is observed among all ONUs; in particular, the maximum latency deviation between ONUs hardly exceeds 1/5 of the upstream frame duration in *FT* scheduling, while *rotation* scheduling totally eliminates the latency discrepancy between ONUs at the expense of a slight average latency increment. A significant latency improvement is also observed when RP redistribution is performed at low network loads as Fig. 5 depicts. Following Fig. 5, the improvement amounts to 80% for the 10 ONU network, while the improvement is less pronounced as the number of ONUs grows to 32. This owes to the fact that the granted excess bandwidth per ONU becomes less, and only a limited number of packets can be sent without being reported. As far throughput is concerned, the figures clearly demonstrate that all ONUs receive an equal amount of resources on average. Moreover, BUDA does not waste bandwidth since the served load equals the requested. Taking Fig. 3*b* for example, the measured throughput increases by 25 Mb/s per 0.1 load increase, which equals the traffic increase per ONU in the 10 ONU network.

The presented results also suggest that BUDA maintains the fairness aspects of FQ and that ONUs receive a fair treatment both in terms of latency and in terms of throughput. To further investigate this result, we simulated scenarios in which the load between ONU is not equal, but separated ONUs into groups of increasing traffic. In the 10 ONU network the ONUs were separated into two groups (high and low traffic), and the high-traffic ONUs produced twice the traffic of their low-traffic counterparts. The simulation results for this scenario results are presented in Figs. 6*a* and *b*, where it is shown that BUDA fully adapts to the ONU traffic without wasting resources. Assuming a total load of 90%, the total expected throughput amounts to 2.25 Gb/s, with high-traffic ONUs receiving 300 Mb/s each and low-traffic ones receiving 150 Mb/s. Similar results are obtained for the 32 ONU network, where four traffic intensities are considered (Figs. 6*c* and *d*,). In this scenario the second, third and fourth ONU groups produced two, three and four times more traffic, respectively, than the first low-traffic group. It is straightforward to verify that bandwidth assignment to ONU groups is performed in increasing order of demand, while no waste of resources is observed.

With respect to latency, it is observed from Figs. 6*a–c* that high-traffic ONUs experience more delay that their low-traffic counterparts. This result verifies that BUDA is fair in the FQ sense, which mandates that (i) ONUs are served in an increasing order of demand, and (ii) ONUs receive the bandwidth they have requested *on average* and not on *a per upstream* frame basis. In fact, requests from higher-traffic ONUs are only partially served during busy upstream frames and the transmission of the remaining request is postponed by BUDA for the next frame. The practical implication of this result is that BUDA does take into account the user subscription level and users with a better SLA (grouped at lower traffic ONUs) enjoy lower latencies. Moreover, if an ill-behaved ONU tries to monopolise the upstream capacity then it simply punishes itself with additional latency, since BUDA

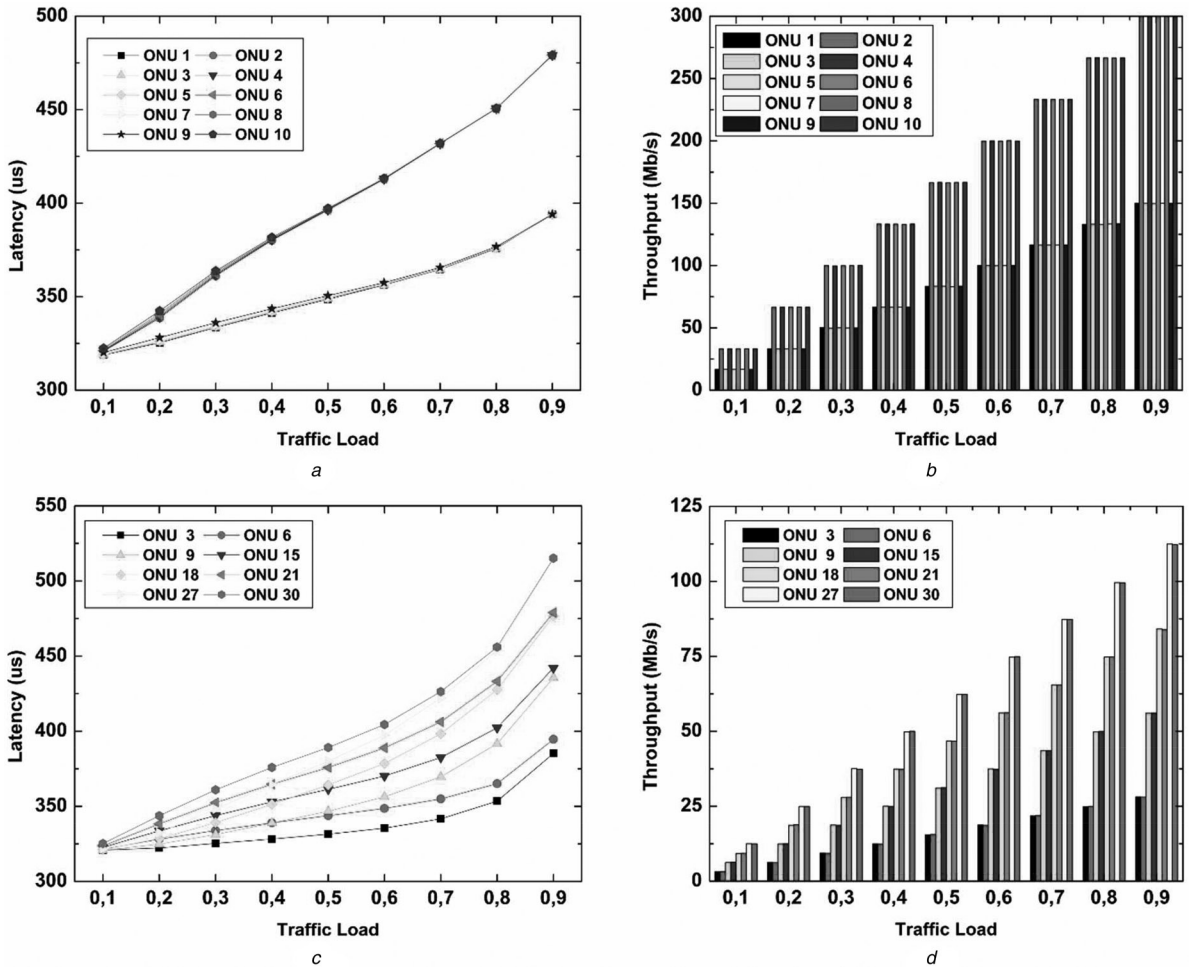

Fig. 6 *BUDA performance when ONUs are grouped in traffic groups*

a BUDA FQ-spatial latency (10 ONUs and two traffic groups)
b BUDA FQ-spatial throughput (10 ONUs and two traffic groups)
c BUDA FQ-spatial latency (32 ONUs and four traffic groups)
d BUDA FQ-spatial throughput (32 ONUs and four traffic groups)

will always prioritise the service of well-behaved ONUs. It should also be noted that not all DBAs will necessarily maintain fairness; in a polling algorithm like IPACT, for example, an ONU receives service once per polling cycle and as a result the average latency is similar for all ONUs irrespective of their load.

Finally, the latency performance of BUDA was also compared with IPACT, a well-studied E-PON DBA, and the efficient bandwidth utilisation (EBU) DBA, which was recently proposed for application in XG-PONs [18]. With respect to IPACT, two variations were implemented: the limited service (IPACT-limited), where ONUs are granted with their request unless it exceeds their share of the bandwidth

$$W_t^n = \min\{R_t^n, \mathrm{BW}/m\}, \tag{7}$$

and the gated service (IPACT-gated), where ONUs are always granted with their request

$$W_t^n = R_t^n \tag{8}$$

The comparison between IPACT and BUDA is illustrated in Fig. 7 and it becomes evident that the BUDA variants outperform IPACT-limited and perform equally well with IPACT-gated in terms of latency, while no notable difference is observed in the total throughput. The relative latency benefit of BUDA FQ-Spatial, in particular, amounts to 20% and 7.5% in comparison with IPACT-limited and gated, respectively, in the 10 ONU topology, while for the 32 ONU network the latency gain increases to 68% and 9.5%, respectively. Similar results are observed when the RP redistribution is applied, although BUDA in addition exhibits lower latency than IPACT in low loads and a 32 ONU network.

As far as the EBU, is concerned, this DBA resembles the elastic service version of IPACT and aims to distribute bandwidth to ONUs within a SI period that endures an integer number of upstream frames (multiple of 125 μs). During each SI, an ONU can receive a maximum number of allocated bytes (AB), thus the average throughput of each ONU enjoys equals(AB/SI), while a peak throughput *SI* times greater than the average can also be achieved in an upstream frame, assuming that only a single ONU has requested bandwidth. EBU utilises two down counters to monitor the available bytes that can be allocated to ONUs within the SI (counter *VB*), and the remaining duration of the SI itself (counter *SI_timer*). Each ONU is allocated with either the bytes it requests per upstream frame, or the *AB* value (in particular the minimum between these values), provided that (i) the *VB* counter of the corresponding ONU is greater than zero within the *SI*, and (ii) the current upstream is not fully occupied from the requests of other ONUs. According to the above allocation procedure, the VB counter may result into negative values, meaning that the ONU allocated with additional, probably, unutilised bandwidth, and that defines the novelty of EBU. Whenever the *SI_timer* expires, the *VB* counter of the corresponding ONU must be re-initialised and the new *SI* commences. In general the *VB* counter is re-charged to the *AB* value. However, in case an ONU has been allocated with excess bandwidth (negative *VB* counter) that the reminder ONUs

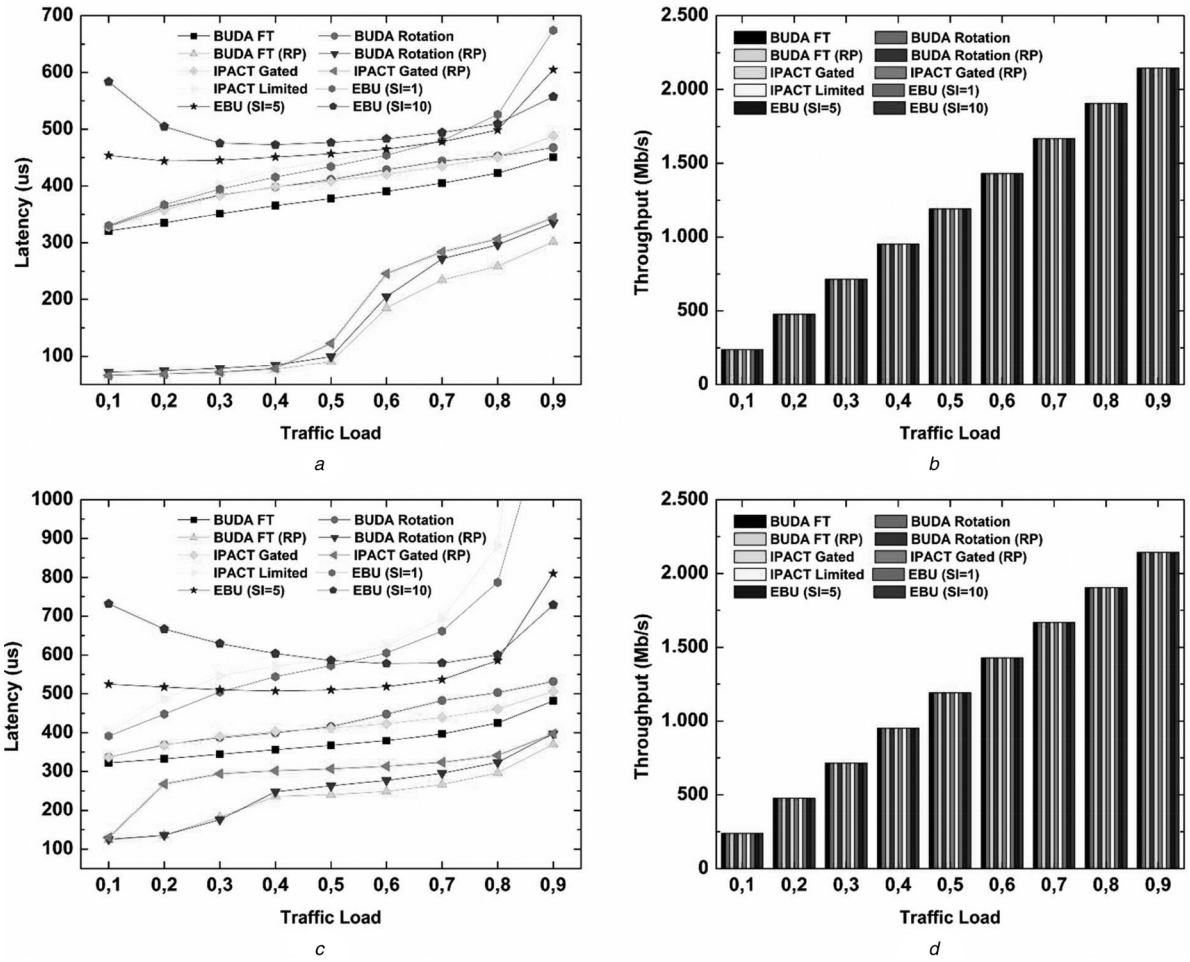

Fig. 7 *Average latency and throughput comparison of BUDA, IPACT and EBU*

a Average latency (10 ONUs)
b Average throughput (10 ONUs)
c Average latency (32 ONUs)
d Average throughput (32 ONUs)

could not cover (the bandwidth was fully utilised), then the difference is subtracted from the *AB* value and the reminder is for the corresponding VB counter to be re-charged with.

EBU adapts bandwidth allocation to the instantaneous ONU traffic pattern and it is expected that an increasing *SI* value will yield better results, since ONUs have potentially more options to transmit their data before counters reset. Unfortunately, if an ONU does not have bandwidth request at any point of the *SI*, then EBU considers it idle and it may not request more bandwidth before the SI expires. This means that very large *SI* values should be avoided because ONUs will starve at low loads. On the other hand, *SI* values close to unity offer very limited flexibility to the bandwidth allocation process and EBU reverts to a resource allocation scheme that resembles IPACT-limited for *SI* equal to one. In our implementation *SI* was set equal to 1, 5 and 10 frames, so as to present results for both short and long intervals, while the *AB* values where calculated from

$$\mathrm{AB} = \mathrm{SI} * \frac{\mathrm{BW}}{m}. \tag{9}$$

The EBU dependence on the selection of the *SI*, as well as its comparison with BUDA, is summarised in Fig. 7. For a small *SI* value, the EBU latency bears close resemblance to IPACT-limited one, but EBU does not compare well with BUDA irrespective of the *SI* selection. This is especially true at low loads, where ONUs are often classified as idle by the EBU for the whole duration of the *SI* and therefore have to wait unnecessarily. Is should be noted that the EBU performance would be improved if constant-rate traffic such as telephony was introduced in the network, since in this scenario ONUs are never idle, but this study is beyond the scope of this work.

6 Conclusion

We presented an XG-PON specific DBA algorithm that achieves low latency and maintains fairness in the XG-PON. The proposed algorithm utilises the FQ principle to successfully implement burst-by-burst scheduling among different ONUs irrespective of their spatial position. The fairness properties of the algorithm are validated via simulation and it is demonstrated that ONUs are served in order of increasing demand, and that all ONUs are fully served with the bandwidth they request. Simulation results also show that the proposed algorithm outperforms the IPACT DBA in terms of fairness and the EBU DBA in terms of latency.

7 Acknowledgments

This work has been funded by the NSRF (2007-2013) Synergasia-II/EPAN-II Program 'Asymmetric Passive Optical Network for xDSL and FTTH Access,' General Secretariat for Research and Technology, Ministry of Education, Religious Affairs, Culture and Sports (contract no. 09SYN-71-839).

8 References

1 Effenberger, F., Cleary, D., Haran, O., *et al.*: 'An introduction to PON technologies', *IEEE Commun. Mag.*, 2007, **45**, (3), pp. S17–S25
2 Bianco, A., Bonald, T., Cuda, D., *et al.*: 'Cost, power consumption and performance evaluation of metro networks', *IEEE/OSA J. Opt. Commun. Netw.*, 2013, **5**, (1), pp. 81–91
3 Wong, E.: 'Next-generation broadband access networks and technologies', *IEEE/OSA J. Lightwave Techn.*, 2012, **30**, (4), pp. 597–608
4 Bertsekas, D., Gallager, R.: 'Data networks' (Prentice- Hall, 1992)
5 IEEE Standard for Information technology--Local and metropolitan area networks--Specific requirements–Part 3: CSMA/CD Access Method and Physical Layer Specifications Amendment 1: 'Physical Layer Specifications and Management Parameters for 10 Gb/s Passive Optical Networks – 802.3av', 2009. Available at http://ieeexplore.ieee.org/servlet/opac?punumber=5294944. (Retrieved: 26 October 2015)
6 Kramer, G., Mukherjee, B., Pesavento, G.: 'Interleaved polling with adaptive cycle time (IPACT): a dynamic bandwidth distribution scheme in an optical access network', *Photonic Netw. Commun.*, 2002, **4**, (1), pp. 89–107
7 McGarry, M.P., Reisslein, M.: 'Investigation of the DBA algorithm design space for EPONs', *IEEE/OSA J. Lightwave Techn.*, 2012, **30**, (14), pp. 2271–2280
8 Mcgarry, M.P., Reisslein, M., Maier, M.: 'Ethernet passive optical network architectures and dynamic bandwidth allocation algorithms', *IEEE Commun. Surv. Tutorials*, 2008, **10**, (3), pp. 46–60
9 Zheng, J., Mouftah, H.: 'A survey of dynamic bandwidth allocation algorithms for Ethernet Passive Optical Networks', *Opt. Switching Netw.*, 2009, **6**, (3), pp. 151–162
10 Assi, C.M., Ye, Y., Dixit, S., *et al.*: 'Dynamic bandwidth allocation for quality-of-service over Ethernet PONs', *IEEE Selected Areas in Commun.*, 2003, **21**, (9), pp. 1467–1477
11 Bai, X., Shami, A., Assi, C.: 'On the fairness of dynamic bandwidth allocation schemes in Ethernet passive optical networks', *Computer Commun.*, 2006, **29**, (11), pp. 2123–2135
12 Byun, H.-J., Nho, J.-M., Lim, J.-T.: 'Dynamic bandwidth allocation algorithm in Ethernet passive optical networks', *Electron. Lett.*, 2003, **39**, (13), pp. 1001–1002
13 Luo, Y., Ansari, N.: 'Bandwidth allocation for multiservice access on EPONs', *IEEE Commun. Mag.*, 2005, **43**, (2), pp. S16–S21
14 Kramer, G., Banerjee, A., Singhal, N.K., *et al.*: 'Fair queueing with service envelopes (FQSE): a cousin-fair hierarchical scheduler for subscriber access networks', *IEEE Sel. Areas Commun. J.*, 2004, **22**, (8), pp. 1497–1513
15 ITU-T (G.987): '10-Gigabit-capable passive optical network (XG-PON) systems', 2010. Available at https://www.itu.int/rec/T-REC-G.987/e. (Retrieved: 26 October 2015)
16 Leligou, H.C., Linardakis, C., Kanonakis, K., *et al.*: 'Efficient medium arbitration of FSAN compliant GPONs', *Int. J. Commun. Syst.*, 2006, **19**, pp. 603–617
17 Han, M.-S., Yoo, H., Yoon, B.-Y., *et al.*: 'Efficient dynamic bandwidth allocation for FSAN-Compliant GPON', *OSA J. Opt. Netw.*, 2008, **7**, (8), pp. 783–795
18 Han, M.S., Yoo, H., Lee, D.S.: 'Development of efficient dynamic bandwidth allocation algorithm for XGPON', *ETRI J.*, 2013, **35**, (1), pp. 18–26
19 Han, M.-S.: 'Dynamic bandwidth allocation with high utilization for XG-PON'. Proc. Advanced Communication Technology (ICACT), 2014, pp. 994–997
20 Skubic, B., Chen, J., Ahmed, J., *et al.*: 'A comparison of dynamic bandwidth allocation for EPON, GPON, and next-generation TDM PON', *IEEE Commun. Mag.*, 2009, **47**, (3), pp. S40–S48
21 Gravalos, I., Yiannopoulos, K., Papadimitriou, G., *et al.*: 'The max-min fair approach on dynamic bandwidth allocation for XG-PONs', *Wiley Trans. Emerging Telecommun. Technol.*, 2015, **26**, (10), pp. 1212–1224
22 Effenberger, F.J.: 'The XG-PON system: Cost effective10 Gb/s access', *IEEE/OSA J. Lightwave Techn.*, 2011, **29**, (4), pp. 403–409
23 Parekh, A.K., Gallager, R.G.: 'A generalized processor sharing approach to flow control in integrated services networks: The single node case', *IEEE/ACM Trans. Netw.*, 1993, **1**, (3), pp. 344–357
24 Shreedhar, M., Varghese, G.: 'Efficient fair queueing using deficit round robin'. Proc SIGCOMM, 1995, pp. 231–242
25 Goyal, P., Vin, H.M., Cheng, H.: 'Start-time fair queueing: a scheduling algorithm for integrated services packet switching networks', *IEEE/ACM Trans. Netw.*, 1997, **5**, (5), pp. 690–704
26 Bennett, J.C.R., Zhang, H.: 'WF2Q: worst-case fair weighted fair queueing'. Proc INFOCOM, 1996, pp. 120–128
27 Golestani, S.J.: 'A self-clocked fair queueing scheme for broadband applications'. Proc. INFOCOM, 1994, pp. 636–646
28 Stoica, I., Shenker, S., Zhang, H.: 'Core-stateless fair queueing: a scalable architecture to approximate fair bandwidth allocations in high-speed networks', *IEEE/ACM Trans. Netw.*, 2003, **11**, (1), pp. 33–46
29 Lu, S., Bharghavan, V., Srikant, R.: 'Fair scheduling in wireless packet networks', *IEEE/ACM Trans. Netw.*, 1999, **7**, (4), pp. 473–489
30 Vaidya, N., Dugar, A., Gupta, S., *et al.*: 'Distributed fair scheduling in a wireless LAN', *in IEEE Trans. Mobile Comput.*, 2005, **4**, (6), pp. 616–629
31 Song, H., Kim, B., Mukherjee, B.: 'Multi-thread polling: A dynamic bandwidth distribution scheme in long-reach PON', *IEEE J. Sel. Areas Commun.*, 2009, **27**, (2), pp. 134–142
32 OMNET++ Discrete Event Simulator. Available at http://www.omnetpp.org/. (Retrieved: 26 October 2015)
33 Hurtig, P., John, W., Brunstrom, A.: 'Recent trends in TCP packet level characteristics'. Proc. International Conference on Networking and Services (ICNS), 2011