Low-Latency Upstream Scheduling in Multi-Tenant, SLA Compliant TWDM PON

Arijeet Ganguli and Marco Ruffini
School of Computer Science and Statistics, Trinity College Dublin, Ireland
gangulia@tcd.ie, marco.ruffini@tcd.ie

Abstract: We present a multi-tenant multi-wavelength upstream transmission scheme for virtualised PONs, enabling compliance with latency-oriented Service Level Agreements (SLAs). Our analysis highlights an important trade-off between single-channel vs. multi-channel PONs, depending on ONUs tuning time. © 2023 The Author(s)

1. Introduction

Over the past decade, network architectures have transitioned from closed systems to open and disaggregated systems. Software Defined Networking (SDN) and Network Function Virtualization (NFV) have played a key role, offering programmability, cost efficiency and flexibility. Passive Optical Networks (PONs) have experienced a similar transition towards virtualisation [1, 2], which enables dynamic software-based control of scheduling algorithms and multi-tenancy. This means independent Virtual Network Operators (VNOs) are able to run multiple independent upstream scheduling algorithms in a shared PON.

In order to support 5G and beyond use cases and business applications (including fronthaul for different type of mobile services), which require low latency guarantee, PONs will need to support Service Level Agreements (SLAs). This means that capacity sharing cannot be achieved at the expense of latency performance. This issue has been addressed by work in [2], where DBA algorithms can be modified to suit specific applications, and in [3], with the development of a virtual Dynamic Bandwidth Allocation (vDBA) architecture, which allows tenants to specify indiviual allocation grant with microsecond precision.

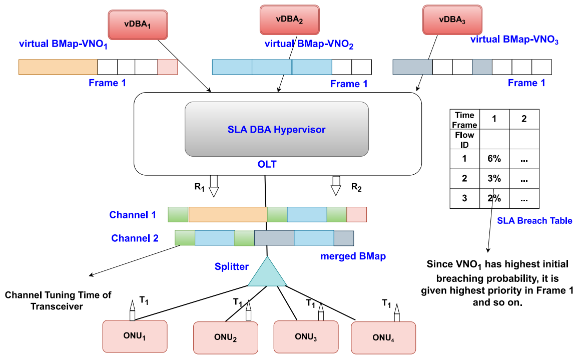
In this work we extend our vDBA concept to a multi-channel PON network, where the merging of Bandwidth Maps (BMaps) is carried out across multiple wavelength channels. The OLT, besides indicating the time at which the ONUs can transmit, it also indicates the wavelength channel. The key objective is to define a number of SLAs, expressed in terms of maximum latency and compliance level, and to propose an algorithm that can operate the Bandwidth Map merging operation, minimising the probability to breach SLAs, across all VNOs.

The evaluation of our work is based on two key performance analysis. First we compare the latency for systems using different line rates and number of channels. Assuming an overall PON capacity of 200Gb/s, we consider a system with 8 channels at 25Gb/s, one with 4 channels running at 50Gb/s, and one with a single channel at 200Gb/s (being single channel there is no multi-wavelength allocation for the 200Gb/s option). Then we show how the difference in performance changes when we consider different tuning times for the ONU transmitters. For this we compare the case of negligible tuning time (i.e., if the system implements channel bonding, [4], so that the ONU has more than one transceiver always ready to transmit at least at another wavelength); the case of Class 1 transmitters (i.e., tuning time $< 10\mu s$) [5], with tuning time of 250 ns (i.e., close to the burst overhead time) and 1 us; and a Class 2 transmitters (i.e., tuning time $> 10\mu s$ but $< 25ms$), with tuning time of 15 us. Class 3 devices are not considered has they have tuning times longer than 25 ms, which is more than two order of magnitudes higher than the PON frame, and would in practice represent a semi-static allocation.

Our results show that our proposed multi-wavelength algorithm is capable of maintaining high percentage of SLA compliance, even for high network load. In addition, we highlight an important trade-off between the latency of multi-channel systems and the transmitters' tuning time. If the tuning time is negligible compared to the burst overhead, the multi-channel system has better latency performance. This is due to the fact that the burst overhead tends to have similar time duration independently of the line rate [6], thus higher line rate channels require a proportionally larger amount of bits than lower line rate channels. However as the ONU tuning time increases, the single channel system outperforms the multi-channel ones.

2. System architecture

The use case and architecture addressed in this work is shown in Fig. 1. In our approach, multiple VNOs run different schedulers in parallel (vDBAs), each forwarding a virtual Bandwidth Map (vBMap), which allocates upstream transmission slots to a group of ONUs. This for example allows multiple mobile operators to run fronthaul services (i.e., 7.2 RAN split), using independent Cooperative Transport Interfaces (CTI) The scheduling hypervisor (or merging engine) collects all such virtual bandwidth maps to create a single physical bandwidth map, resolving any collisions between slots overlapping in time. Such collisions can be resolved by delaying some of the grants (at the risk of breaching SLAs). In this multi-wavelength approach, the OLT also has the option to select transmission over different channels, to minimise grant delay, although this is constrained by the ONU tuning time. Our merging engine makes these decisions based on specific Service Level Agreements (SLAs), so that it minimises the probability of breaching SLAs (which is key for supporting 5G and future 6G services). The



Time Frame	1	2
Flow ID		
1	6%	...
2	3%	...
3	2%	...

Since VNO$_1$ has highest initial breaching probability, it is given highest priority in Frame 1 and so on.

Fig. 1: TWDM multi-tenant upstream scheduling algorithm

Notations :

N : Number of bmaps from the respective VNOs
M : Number of SLAs
W : Number of channels
F : Frame size in allocation units
SLA$_i$: SLA ID for the ith VNO
lat$_i$: Maximum allowed packet level latency for the allocations from the ith VNO
br$_i^{pt}$: Maximum fraction of packets in a flow that can breach packet level SLA
|S$_i$| : Number of allocations in the ith bmap
bwi,j : jth bandwidth allocation report from the ith bmap
req$_{i,j}$: Request time of bw$_{i,j}$
t$_{i,j}^{max}$: Maximum time bw$_{i,j}$ can be delayed without breaching the packet level SLA
$\mathcal{X}_{i,j,k,l}$: Binary Decision Variable, bw$_{i,j}$ alloted time slot l of channel k if equals 1 else 0
$\mathcal{I}$: Boolean truth value function

$$t_{i,j}^{max} = req_{i,j} + lat_i \qquad (1)$$

$$\min \sum_{i=1}^{N} \mathcal{I}\left(\frac{1}{|S_i|} \sum_{j=1}^{F} \sum_{k=1}^{W} \sum_{l=1}^{F} X_{i,j,k,l} \mathcal{I}(l > t_{i,j}^{max}) > br_i^{pt}\right) \qquad (2)$$

$$\text{s.t} \sum_{i=1}^{N} \sum_{j=1}^{F} \mathcal{X}_{i,j,k,l} \leq 1 \;\; \forall\, 1 \leq k \leq W, 1 \leq l \leq F \;\; (3)$$

$$\sum_{k=1}^{W} \sum_{l=1}^{F} \mathcal{X}_{i,j,k,l} = 1 \;\; \forall\, 1 \leq i \leq N, 1 \leq j \leq F \;\; (4)$$

$$\sum_{j=1}^{F} \sum_{k=1}^{W} \sum_{l=1}^{F} \mathcal{X}_{i,j,k,l} = |S_i| \;\; \forall\, 1 \leq i \leq N \;\; (5)$$

Fig. 2: MILP Notations and Equations

use of a stateful algorithm, which takes into consideration the history of a service flow when making scheduling prioritisation decisions, is preferred to stateless algorithms [7]. This is because a stateful algorithm can prioritise flows depending on how close they are to breaching their specific SLA target [8].

Thus in this work, we propose a heuristic stateful TWDM scheduling algorithm. The objective of the algorithm is to maximise the SLA compliance across all the flows during upstream transmission. We focus specifically on the additional latency introduced by the multi-sharing aspect of the PON. An SLA breach occurs when a given flow accumulates a number of delayed upstream slots that is above its target SLA threshold. For example for an SLA with maximum merging delay of 25 μs with 99% compliance, every time an upstream slot is delayed by more than 25 μs with respect to the requested time slot in the virtual BMap, we increment a counter. If the counter goes above the non-compliance rate (in this case 100-99=1%), calculated over a number of frames (i.e., we use a 1ms window, which is the time duration of a 5G sub-frame), then we consider that an SLA breach has occurred.

The problem can be formulated as Mixed Integer Programming (MIP) as shown in Fig. 2. Equation (1) calculates the maximum delay of any given allocation to remain within the target threshold for SLA breach. We maintain a 4 dimensional binary decision variable matrix $\boldsymbol{X}$ where the objective function is explained as follows - the inner truth value function calculates the packet level breach and the outer truth value function calculates the flow level breach across each virtual BMap. Equation (3) is the constraint that any particular channel at any time slot transmits at most 1 BMap allocation. Equation (4) is the constraint that all BMap allocations are alloted unique channel time-slot pairs. Equation (5) checks the conservation of BMap allocations within each virtual BMap.

The algorithm maintains a few key data structures. A flow-breach likelihood table keeps track of how far each traffic flow is from breaching its SLA (i.e., going above non-compliance rate); this is important to minimise SLA breach, as the algorithm will prioritise scheduling for the flows that are closer to breach their SLA. A channel freetime table maintains and updates the earliest free time of each channel over time, i.e., when the channel can be reallocated to a different BMap allocation. We also keep track of the earliest free time of the various transceivers of the ONUs (i.e., depending on the ONU tuning time).

With reference to Fig. 3, the algorithm first calculates the allocation maxtime which is the latest time an allocation can be scheduled within its latency target (code lines 1 to 3). Then, it starts allocating slots to the various allocations according to the time assigned by their originating virtual BMaps (code lines 5 to 6). Next, it resolves collisions (lines 8 to 21) by allocating slots first in increasing order of non-compliance rate (line 18), then increasing order of their maxtimes (line 19) and finally increasing order of their sizes (line 20). Lines 22 to 43 traverse the sorted BMap and allocates channel and scheduling time for the allocations. Lines 23 to 26 first get the allocation ONU id, then use it to get the free times of all the transceivers of the respective ONU, then allocates the earliest free transceiver and receiver for transmission of the BMap allocation. Line 27 checks the channel free time table and allocates the earliest free wavelength for transmission. Lines 29 to 34 check if tuning of the transceiver to a new channel is required. Lines 35 and 36 calculate the earliest transceiver and receiver free times, respectively. Lines 37 to 40 calculate the scheduling time for transmission according to whether tuning is required or not and lines 41 to 43 assign the transceiver id, receiver id and scheduling time memory variables of the BMap allocation object. Finally, the lines 44 to 47 recalculate the non-compliance rate of all the flows and updates the flow-breach table for scheduling of the allocations in the next time frame.

3. Performance Evaluation

In order to set up the simulation environment, we generate input BMaps from different VNOs. The allocation load on the shared PON was considered for 20%, 50% and 80% of the total upstream capacity (i.e., 200Gb/s across all channels considered). For each of this allocation loads, we then varied the percentage allocated to SLA-driven flows from 10% to 100% of the total load (the remaining part is allocated to best effort flows). We consider 5 VNOs

```
Input : bmaps, slaTable, breachTable,
channelFreetimeTable, transFreeTimeTable,
recFreeTimeTable, transChannelMap,
tuningTime, bmapsSize
Output : mergedBmap, transId, recId
for vnoId, bmap in bmaps do
    for bw in bmap do
        bw.maxtime = bw.reqtime +
                slaTable[bw.slaId].latency
mergedBmap = [ ]
for vnoId, bmap in bmaps do
    append(mergedBmap, bmap)
n = mergedBmap.length
for i from 0 to n-1 do
    for j from i+1 to n-1 do
        vnoId1 = mergedBmap[i].vnoId
        vnoId2 = mergedBmap[j].vnoId
        likelihood1 = breachTable
                [vnoId1]
        likelihood2 = breachTable
                [vnoId2]
        maxtime1 = mergedBmap[i].
                maxtime
        maxtime2 = mergedBmap[j].
                maxtime
        size1 = mergedBmap[i].size
        size2 = mergedBmap[j].size
        if likelihood1 < likelihood2
        or likelihood1 == likelihood2
    and maxtime1 < maxtime2
        or maxtime1 == maxtime2
    and size1 < size2 then
            swap(mergedBmap[i],
            mergedBmap[j])
for bw in bmap do
  onuId = bw.onuId
  transFreetimeArr =
      transFreeTimeTable[onuId]
  transId = transFreetimeArr.min().index
  recId = recFreeTimeTable.min().index
  earliestFreeChannels =
channelFreetimeTable.min()
  currentTunedChannel =
transChannelMap[onuId][transId]
  if currentTunedChannel not in
      earliestFreeChannels then
    tuningRequired = True
    selectChannel =
        selectFrom(earliestFreeChannels)
  else
    tuningRequired = False
    selectChannel = currentTunedChannel
  transFreetime = transFreetimeArr
                [transId]
  recFreetime = recFreeTimeTable
                [recId]
  if tuningRequired then
    sched_time = max(transFreetime +
            tuningTime,  recFreetime)
  else
    sched_time = max(transFreetime,
                        recFreetime)
  mergedBmap[i].transId = transId
  mergedBmap[i].recId = recId
  mergedBmap[i].sched_time = sched_time
for bw in bmap do
  if bw.maxtime < bw.schedtime then
      flowBreachVnoId = bw.vnoId
      breachTable[vnoId] +=
                1/bmapsSize[vnoId]
return mergedBmap
```

Fig. 3: Pseudocode for TWDM

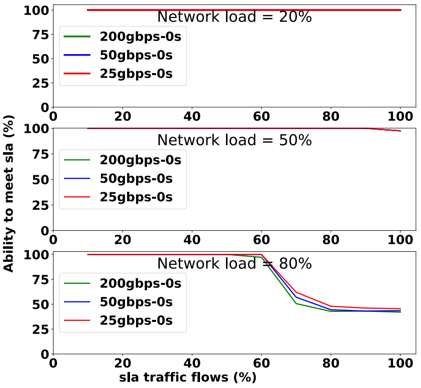


Fig. 4: Tuning time: 0s

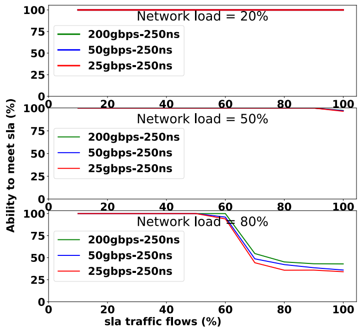


Fig. 5: Tuning time: 250ns

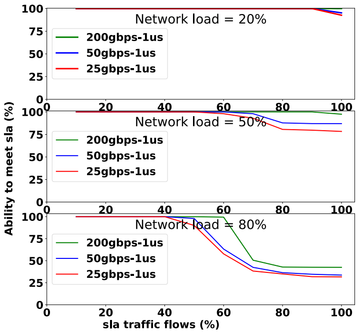


Fig. 6: Tuning time: $1\mu s$

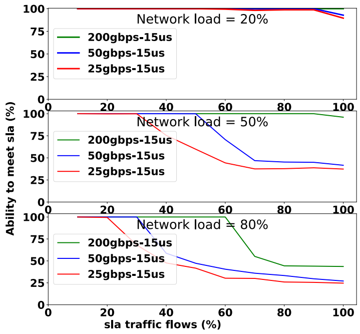


Fig. 7: Tuning time: $15\mu s$

(each one generating a virtual BMap every frame (125 μs)) and 2 types of SLAs: one requiring 90% compliance with a latency target of 12.5 μs and one requiring 95% compliance with a latency target of 25 μs. Each bandwidth map has a uniformly distributed set of allocations, with average burst size set at 6% of the total frame size for a 25Gb/s line rate (the same ONU is also allowed to provide multiple burst per frame). An empty time slot of 0.33 μs, is introduced between allocations to account for guard time between upstream transmissions. Each ONU has one transceiver, which is tunable on any of the available channels, and constrained by a tuning time, which is a simulation parameter. As mentioned above, the system under investigations, are 8 x 25Gb/s, 4 x 50Gb/s and 1 x 200Gb/s channels. The tuning times considered span from negligible to 250 ns, 1 μs and 15 μs.

The experiment was run for 5000 time frames and the average number of SLA breaches is recorded. The results show the ability of the system to comply with the SLAs (y axis) versus the percentage of total flows that require SLA. The different plots in the same figure consider PON loads of 20%, 50% and 80% of the total capacity.

The plot in Fig. 4 reports the ability to meet SLA for negligible channel tuning time. We can see that up to 50% load there is always full compliance (except a slight drop for SLA flows above 90%). For 80% load, compliance starts dropping from 60% of SLA traffic for the 8X25G and 4x50G systems and from 50% for the 1x200G. As mentioned above the multi-channel approaches outperform the single channel as at 200G rate the burst overhead is proportionally larger (i.e., in terms of bits required). As the tuning time increases, in Fig. 5, 6 and 7, we see that the latency performance of the multi-channel system decreases, as the algorithm progressively looses the ability to avoid scheduling delays by tuning on different channels.

4. Conclusions

In this work, we presented a heuristic stateful algorithm for upstream scheduling in a TWDM multi-tenant PON, capable of satisfying SLAs. We show the role that ONU tuning time plays in terms of latency performance in the trade-off between higher capacity single channel and lower capacity multi-channel systems. While we believe that there is room for improvement in multi-channel scheduling algorithms for longer tuning times, the current results show that tuning times between 250 ns and 1 μs would be required to maintain satisfactory performance in multi-tenant, multi-channel PON systems.

Acknowledgments. Support from SFI grants 12/RC/2276_p2, 17/CDA/4760 and 13/RC/2077_p2 is acknowledged.

References

1. G. Simon et al. MEC and Fixed Access Networks Synergies. OFC'21
2. J-I. Kani et al., Flexible Access System Architecture to Support Diverse Requirements and Agile Service Creation. JLT, April 2018
3. M. Ruffini et al. The Virtual DBA: Virtualizing Passive Optical Networks to Enable Multi-Service Operation in True Multi-Tenant Environments. JOCN, No.4, Vol.12, April 2020.
4. L. Zhang et al. Channel bonding design for 100 Gb/s PON based on FEC codeword alignment. OFC'17.
5. ITU-T G.989.2 – 40-Gigabit-capable passive optical networks 2 (NG-PON2): PMD layer specification, Feb. 2019.
6. ITU-T G.9804.3 50-Gigabit-capable passive optical networks (50G-PON): PMD layer specification. Sep. 2021.
7. F. Slyne et al. Stateful DBA Hypervisor Supporting SLAs with Low Latency High Availability in Shared PON. OFC'21.
8. A. Ganguli et al. Real-time, low latency virtual DBA hypervisor for SLA-compliant multi-service operations over shared Passive Optical Networks. OFC'23.