Fair Queuing with Service Envelopes (FQSE): a cousin-fair hierarchical scheduler for Ethernet PONs

G. Kramer, A. Banerjee, N. K. Singhal, and B. Mukherjee
Department of Computer Science, University of California – Davis, Davis, California 95616
{gkramer,abanerjee}@ucdavis.edu, {singhaln,mukherje}@cs.ucdavis.edu

S. Dixit and Y. Ye
Nokia Research Center, Burlington, Massachusetts 01083
{sudhir.dixit,yinghua.ye}@nokia.com

Abstract: Non-hierarchical algorithms are not scalable for scheduling in EPON, while hierarchical algorithms do not provide fairness among the end consumers. This study presents a novel hierarchical EPON scheduler, which guarantees fairness among all subscribers.

1. Introduction

Ethernet Passive Optical Networks (EPON) are point-to-multipoint optical access networks. An Optical Line Terminal (OLT) at the Central Office (CO) is connected to many Optical Network Units (ONUs) at remote ends using optical fiber and passive splitter [1]. Each ONU acts as an access point for multiple subscribers, each of whom has one or more queues. Since the ONUs share the same optical channel for upstream transmission, the OLT must schedule timeslots for the ONUs to transmit data. The timeslot assignment is facilitated by using GATE and REPORT control messages to grant and request bandwidth respectively.

A fundamental requirement for an access network is to be able to guarantee minimum service (bandwidth) to each subscriber and ensure fairness in distributing excess bandwidth, if any is available. One way to achieve this is to assign timeslot to each subscriber independently. However, such method is not scalable with the number of subscribers as it requires a separate GATE and REPORT message to each subscriber, and given a large number of subscribers, may consume a significant portion of EPON bandwidth.

Alternatively, EPON may employ a hierarchical scheduling scheme, where central scheduler in the OLT assigns timeslots to individual ONUs. In turn, each ONU subdivides its assigned timeslot among the subscribers connected to it. Such hierarchical schemes are free from the scalability issues mentioned above, however, they cannot guarantee fairness among the subscribers. Most hierarchical scheduling algorithms that we surveyed in literature, allow fairness only between siblings (i.e., nodes having the same parent). We call these schemes *sibling-fair*. A *sibling-fair* algorithm may discriminate between users connected to different ONUs. Hence, there is a need for a *cousin-fair* scheduler, which schedules fairly to leaf nodes (subscribers) in the hierarchy. Figure 1 illustrates the difference between *sibling-fair* and *cousin-fair* schedulers. The weights of the queues and the queue sizes are denoted by φ and q respectively and w denotes the size of the scheduled slot in bytes. In a *sibling-fair* scheduler in Figure 1(a), fairness is ensured between the ONU A and ONU B, and between the queues in each ONU, but not among queues belonging to different ONUs. Therefore, queues 1 and 3 get unequal slots, although they have the same weight. In a *cousin-fair* scheme, in Figure 1(b), queues 1 and 3 get equal slots.
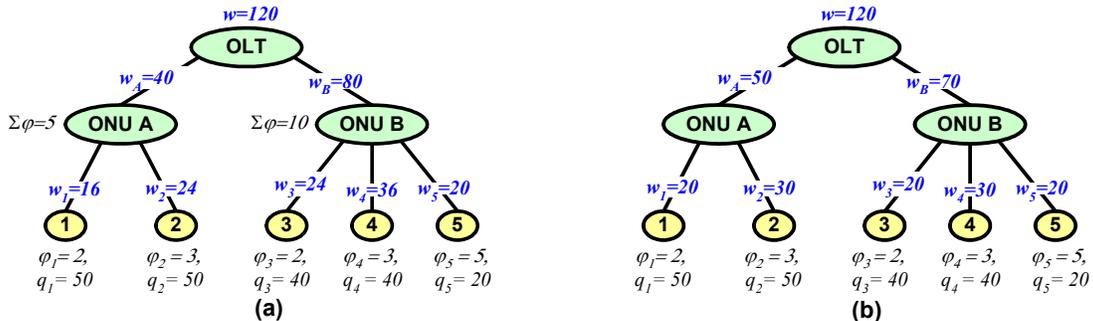

Fig. 1. Comparison between a two level (a) *sibling-fair* and (b) *cousin-fair* scheduler.

Thus, to avoid the scalability issues, the scheduling algorithm should be hierarchical, and to achieve fairness it should be cousin-fair. We found, that out of the vast number of scheduling algorithms surveyed in literature, none

of them meet both requirements. In this study, we present FQSE - a novel algorithm that successfully achieves both goals: it is hierarchical and cousin-fair.

2. Fair Queuing with Service Envelopes (FQSE)

To provide a reasonable level of QoS, we define each queue i to be guaranteed a minimum bandwidth B_i^{MIN}. In EPON, bandwidth is assigned in timeslots. Thus, it is convenient to define minimum guaranteed timeslot size W_i^{MIN} that should be given to a queue i to guarantee its minimum bandwidth B_i^{MIN} ($W_i^{MIN} = B_i^{MIN} T$, where T is called *cycle time*). In addition, each queue has a weight φ_i assigned to it. The values of B_i^{MIN} and φ_i may be varied to provide different levels of service to different users. The objective of the scheduler is to guarantee the minimum bandwidth to each queue, and fairly (i.e., proportionally to weights) share the excess bandwidth.

Consider the kth cycle of bandwidth allocation. The length of a queue i at the beginning of cycle k is defined as $q_{i,k}$. We define the actual amount of bandwidth that queue i gets in the kth cycle as $w_{i,k}$. The minimum slot size that must be granted to queue i in cycle k is defined as $w_{i,k}^{MIN}$:

$$w_{i,k}^{MIN} = \min\left\{ q_{i,k}, W_i^{MIN} \right\} \qquad (1)$$

FQSE is based on a concept of a *service envelope* (SE). SE represents the timeslot size given to a node as a function of some non-negative value called *satisfiability parameter* (SP). SP is a measure of how much the demand for bandwidth can be satisfied for a given node (queue, ONU) in the hierarchy. Each node has its associated SE function. We distinguish the construction of a service envelope for a *leaf node* (denoted E^*) from the construction of a service envelope for an *intermediate node* (denoted E).

E^* is a piecewise-linear function consisting of at most two segments (see Figure 2, plots E_1^*, E_2^* and E_3^*). The first segment begins at a point with coordinate $\left(0, w_{i,k}^{MIN}\right)$ and ends at $\left((q_{i,k} - w_{i,k}^{MIN})/\varphi_i, q_{i,k} \right)$. The ending SP value is chosen such that the slope of the first segment is exactly φ_i. The second segment has a slope equal to 0 and continues to infinity. Intuitively, the meaning of the E^* function should be clear: as the satisfiability parameter changes, the E^* function determines the fair timeslot size that should be granted to a queue. In the worst case (when SP = 0), exactly a $w_{i,k}^{MIN}$-byte slot shall be assigned.

SE E_i for an intermediate node i, is built as a sum of SEs of all the node's children:

$$E_{i,k} = \sum_{j \in D_i} E_{j,k} \qquad (2)$$

where D_i is a set containing all children of node i. This is illustrated in Figure 2, plot E_4.

The FQSE scheduling algorithm consists of alternate *requesting* and *granting* phases.

Requesting Phase: At the end of transmission in a previously-assigned timeslot, a *leaf node* should generate a new SE and send it to its parent in a request message. After collecting SEs from all its children, an *intermediate node* would generate its own SE by summing all received envelopes and send it to its parent. In the context of EPON, each queue generates its SE. The ONU generates a SE from the SE's of its queues. This SE is transmitted to the OLT in a REPORT message. However, the REPORT message has a fixed size of some bytes and it may not be possible to encapsulate the SE in it. In this case the SE is approximated to contain at most K points (using techniques described in [3]). These K points may then be transmitted in a REPORT message. At the end of a *requesting phase*, the *root node* (OLT for EPON) receives SEs from all its children, and calculates its own SE using Eq. 2. We define this root SE as $E_{0,k}$.

Granting phase: The *root node* knows the total number of bytes that can be transmitted in one cycle (W_{cycle}). When the root scheduler obtains $E_{0,k}$ in cycle k, it calculates the SP s_k by solving $E_{0,k}(s_k) = W_{cycle}$. Knowing the cycle start time and the SP s_k, the *root node* calculates the timeslot start time $t_{j,k}$ for each child (ONU) $j \in D_0$ from the respective SEs, such that the transmissions from any child do not overlap. s_k and $t_{j,k}$ are then transmitted to each child in GATE messages. Note, that the OLT only needs one GATE message for each ONU. Following this sequence of steps in a hierarchical fashion, the leaf node ultimately receives its timeslot start time and s_k. The size of the timeslot can then be computed by the leaf node as $w_{i,k} = E_{i,k}^*(s_k)$. Thus, in the context of EPON, each queue will know the amount of data it may transmit in cycle k.

3. Performance study

We simulate FQSE for an EPON access network consisting of an OLT and 16 ONUs and each ONU containing *64* queues. Distances between ONUs and the OLT are random uniformly distributed over the interval [10 km, 20 km]. The line rate is 1 Gbps, as defined in the objectives of IEEE802.3ah task force [2]. We designate four queues, 1 to 4, in two ONUs, ONU$_A$ and ONU$_B$ as test queues. These queues are assigned different B_i^{MIN} and φ_i, to simulate

different service requirements. Queues 1 and 2 are used to simulate best-effort services with $B_1^{MIN} = 0$ and $\varphi_i = 2$ and $B_1^{MIN} = 0$ and $\varphi_i = 1$. When both queues 1 and 2 are backlogged, we would expect queue 2 to get half the bandwidth compared to queue 1. Queue 3 is assigned $B_1^{MIN} = 10$ Mbps and $\varphi_i = 0$ to simulate constant bit rate service. Queue 4 is assigned $B_1^{MIN} = 10$ Mbps and $\varphi_i = 1$. The rest of the queues were used to generate background traffic (ambient load). Among these background traffic queues, 18 queues in each ONU were assigned $B_i^{MIN} = 1$ Mbps and $\varphi_i = 1$, and the remaining queues were best-effort queues ($B_i^{MIN} = 0$) and $\varphi_i = 1$. Each test queue was input a bursty traffic at average load of 90 Mbps. The traffic was generated using the approach described in [4] and was verified to be self similar with *Hurst parameter*=0.8.
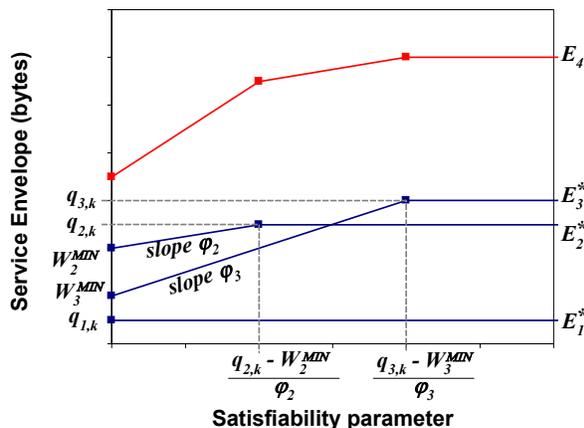

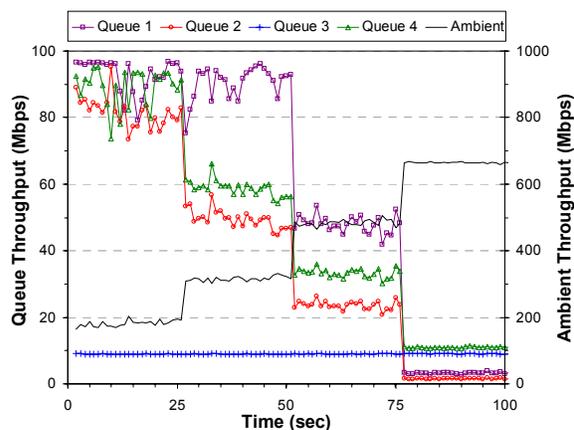Fig. 2. Construction of Service Envelops (SEs)

Fig. 3. Throughput of test queues under different ambient loads

Figure 3 shows the results of our simulations. We note that queue 3, which was configured to have 10 Mbps fixed bandwidth, indeed, has constant throughput irrespective of ambient load. In the first 25-second interval, the ambient load was kept relatively low (~180 Mbps), so that the queues with non-zero weights (queues 1, 2, and 4) were able to send all arrived packets and never became backlogged. When, at time $t = 25$, the ambient load increases to ~320 Mbps, we observe that queues 2 and 4 are not able to transmit all the incoming packets and become backlogged. From this moment on, they maintain the fair relative throughput, with queue 4 always being able to send 10 Mbps more than queue 2, as they were configured to be. Queue 1 is configured to have twice the throughput of queue 2. This, however, would give queue 1 more than 90 Mbps bandwidth, so it only uses 90 Mbps and does not become backlogged until time t=50. At t=50, when the ambient load increases to ~500 Mbps, all four queues become backlogged and all are assigned fair bandwidth. Finally, at time t=75, the ambient load increases even more (to ~660 Mbps) and the available excess bandwidth decreases to a very small amount. At this time, the throughput of each queue approaches its guaranteed bandwidth, 10 Mbps for queues 3 and 4 and zero for queues 1 and 2. Thus, we observe that all queues are assigned bandwidths in a fair manner, in accordance to their configurations.

4. **Conclusion**

In this work, we proposed FQSE − a novel scheduling algorithm for EPON based on a concept of *service envelope* which is a function of *satisfiability parameter*. Our simulations on a sample EPON network show that FQSE meets the dual objectives of being both hierarchical and cousin-fair.

References:
[1] G. Kramer, B. Mukherjee, and A. Maislos, "*Ethernet Passive Optical Networks*", In S. Dixit, editor, "IP over WDM: Building the Next Generation Optical Internet", John Wiley & Sons, Inc., February 2003.
[2] IEEE 802.3ah Ethernet in the First Mile (EFM) Task Force website at "http://www.ieee802.org/3/efm/"
[3] G. Kramer, B. Mukherjee, N. Singhal, A. Banerjee, S. Dixit, and Y. Ye, "*Fair Queuing with Service Envelopes (FQSE): a Cousin-Fair Hierarchical Scheduler and its application in Ethernet PON,*" Technical report # CSE-2003-6, Department of Computer Science, University of California, Davis, CA, March 2003.
[4] M. S. Taqqu, W. Willinger, and R. Sherman, "*Proof of a fundamental result in self-similar traffic modeling,*" ACM/SIGCOMM Computer Communication Review, vol. 27, pp. 5-23, 1997.