International Journal of Wireless Networks and Broadband Technologies
Volume 9 • Issue 2 • July-December 2020

Enhanced Priority Load-Aware Scheduling Algorithm for Wireless Broadband Networks

Aminu Mohammed, Usmanu Danfodiyo University, Sokoto, Nigeria
Abdulhakeem Abdulazeez, Usmanu Danfodiyo University, Sokoto, Nigeria
Ahmed Tambuwal Yusuf, Usmanu Danfodiyo University, Sokoto, Nigeria

ABSTRACT

WiMAX, one of the emerging wireless broadband networks, was designed to support traffic from applications with diverse QoS requirements. In WiMAX, an efficient resource management technique such as scheduling is required for the proper allocation of network resources to these data streams. This article proposes an enhanced priority load-aware scheduling (EPLAS) algorithm to improve the performance of WiMAX networks. The proposed scheme adaptively determines the weight of each queue based on the queue load. It also introduces a packet drop control mechanism that reduces the packet drop rate and increases the average throughput of the network by prioritizing packets with the earliest deadlines within each queue. The performance of EPLAS was evaluated against other benchmark schemes using several simulation experiments. The results revealed that EPLAS performed significantly better than the benchmark algorithms in terms of average delay, average packet drop ratio, and average throughput.

KEYWORDS

Enhanced PLAS, EPLAS, PLAS, Priority Load-Aware Scheduling, Scheduling, Wimax, Wireless Broadband Networks, WRR

INTRODUCTION

Recently, there have been significant advances in wireless broadband technologies to meet the growing demand for high-speed delivery of multimedia services. WiMAX is one such technology designed to provide high-speed internet access over a metropolitan area with a 15-km radius at about 70 Mbps (Nie, Wang & Pack, 2012). Its low cost of installation and flexibility have made it adoptable not only by small businesses but also by residential users. The technology presents some specifications at both the media access control (MAC) and the physical (PHY) layer of the network reference model. At the PHY layer, WiMAX uses orthogonal frequency division multiplexing (OFDM) (Rajeem & Fernando, 2010). OFDM removes delay spread, inter symbol interference, and multi-paths from communication channels to enable speedy transmission of multimedia services. On the other hand, the MAC layer supports quality of service (QoS) classes for proper utilization of network resources

DOI: 10.4018/IJWNBT.20200701.oa1

(Wu, Huang & Huang, 2012). The MAC layer classifies traffic into different classes based on QoS requirements such as delay, bandwidth, jitter, latency, and throughput. To guarantee these requirements and ensure efficient utilization of the often scarce network resources among these classes, a scheduling algorithm is required.

Scheduling is a technique used for the sharing of network resources among competing subscriber stations (SSs). It controls bandwidth allocation and determines the order by which packets are transmitted from different classes (Chin-Ling & Cheng-Yi, 2012). Several scheduling algorithms have been proposed for resource management in WiMAX (Nie et al., 2011; Ahmad, Hamma, & Nasir, 2019; Naik, Dora, & De, 2019). Priority load-aware scheduling (PLAS) is one such algorithm and was designed to provide QoS requirements for each class. It employs a mechanism that prioritizes real-time traffic over non-real-time traffic. The algorithm introduces a dynamic weight, which is computed according to load of each class. It computes and multiplies a priority value and the load-aware weighted round-robin (LAWRR) weight of each class (Saidu, Subramaniam, & Jaafar, 2014). The resultant value becomes the priority weight of the class. The weight value allows the scheduler to serve more packets from the real-time traffic than other traffics in every service round. However, the excess weights allocated to non-real-time traffic cause an increase in the delay of real-time traffic under heavy and equal burst traffic. Also, because packets are served in round-robin (RR) fashion and since the real-time traffics have low tolerance for delay (Mohammed, Saidu & Abdulazeez, 2018), PLAS will increase packet loss and decrease average throughput due to its failure to prioritize packets within each queue according to their deadlines.

In this paper, the authors propose a new scheduling scheme, enhanced priority load-aware scheduling (EPLAS) as an extension of PLAS to improve network performance. The EPLAS algorithm adaptively computes queue weight values and employs packet drop control mechanism to reduce both delay and packet drop rate and to increase throughput of real-time traffic. The performance of EPLAS is evaluated against LAWRR, PLAS, and CBS using simulation experiments. The results reveal that EPLAS performs better in terms of delay, packet drop rate, and throughput.

The rest of this paper is organized as follows: In the Related Works section, an overview of some related scheduling schemes in WiMAX is presented; Problem Definition section, presents the problem, Proposed Algorithm section, describes the proposed EPLAS algorithm; Simulation Result section, presents the simulation results; and the Conclusion section, concludes the paper.

RELATED WORKS

Several scheduling and resource allocation schemes have been proposed for different broadband networks (Iaad, Mustapha, Taoufik, Samer, & Xavier, 2019). Comsa et al. (2018) proposed a smart scheduling scheme to improve the performance of 5G networks. It guarantees the varying QoS requirements different traffics. The scheme employs reinforcement learning and neural network to determine the resource allocation and service order of the contending traffics. The schemes in Comsa et al. (2019a); Comsa et al. (2019b); Comsa De-Domenico and Ktenas (2019); and Comsa et al. (2020) were also proposed for resource allocation in 5G networks. Some of the scheduling schemes proposed for LTE networks includes: Avocanh, Abdennebi, and Ben-Othman (2014); Khan, Martini, Bharucha, and Auer (2012); Zou, Trestian, & Muntean, (2013a), and Zou, Trestian, and Muntean (2013b). This research work however, focuses mainly on WiMAX, therefore we present a review of some of the related scheduling algorithms in WiMAX networks as follows.

The RR algorithm (Hahne, 1991) was proposed to separate traffic streams according to their priorities. RR separates traffic into different queues based on their QoS requirements. It serves packets from all queues moving from high- to low-priority queues in a cyclic manner. The algorithm repeats the same process until all packets are served from all the queues. RR allocates equal resources to all queues. Therefore, it is fair for traffic streams with the same QoS requirements. However, it causes an

increase in delay and packet drop and a decrease in throughput due to its failure to prioritize packets based on their QoS requirements.

Weighted round-robin (WRR) algorithm was proposed by Ketevenis, Sidiroipoulos, and Courcoubetis (1991) to address the lack of QoS guarantees in RR. WRR classifies packets into separate classes based on their QoS priority. The algorithm assigns a static weight counter to all non-empty queues. The weight determines the number of packets that the scheduler can serve from each queue by the end of a counter reset. Packets are served from all queues in an RR fashion, starting from high-priority queues and moving to low-priority queues. The weight counter value of a queue is decremented by one each time a packet is served from that queue. The scheduler continues in this manner until the weight counter values of all the queues are reduced to zero or when the queues become empty. Therefore, WRR is fair when all the queues have equal packet load. However, it will cause an increase in delay and packet loss and a decrease in throughput under high-input traffic as a result of its failure to adjust and prioritize weights according to their QoS requirements.

Mardini and Alfool (2011) proposed a modified weighted round-robin (MWRR) scheduling algorithm to reduce delay in WRR. MWRR computes and assigns weights to each non-empty queue based on its size. The weight is obtained as a product of the static WRR and a dynamic multiplier value. The multiplier, which is meant to increase the service rate of each queue, is an integer value determined by the size of a queue. That is, the larger the queue size, the larger the value and vice versa. MWRR reduces average delay but will lead to an increase in packet drop and a decrease in average throughput of real-time traffic due to its failure to prioritize packets based on their QoS requirements.

A priority weighted round-robin (PWRR) scheduling algorithm was proposed by Manirabona, Boudjit, and Fourati (2016), to reduce delay in real-time traffic. The algorithm classifies and queues packets based on their QoS requirements into real-time (RT) and non-real-time (NRT). PWRR uses WRR to schedule the NRT queues. The algorithm serves the resultant packets from the WRR scheduling process and the packets from the RT queue using priority scheduling (PS). PS prioritizes the RT queue over the NRT. That means packets from the WRR process are served only when there are no packets in the RT queue. The PWRR reduces delay and increases the throughput of RT connections. However, it starves the NRT classes under high-input traffic due to the PS used.

Saidu et al. (2014) proposed an LAWRR algorithm to also reduce delay in WRR. LAWRR dynamically assigns weights to each non-empty queue. At the beginning of each service round, the dynamic weight is computed as a product of a queue's dynamic coefficient and the static weight from the WRR process. The dynamic coefficient is dependent on the load of each queue. That is, the higher the load, the higher the dynamic coefficient of that queue and vice versa. LAWRR reduces average delay and packet loss and increases average throughput. However, it increases delay of real-time traffic due to its failure to prioritize packet transmission according to QoS.

A PLAS algorithm is proposed by Mohammed et al. (2018) to mitigate the problem of delay in LAWRR. PLAS classifies and queues packets based on their QoS requirements and deploys a priority mechanism that prioritizes RT traffic over other traffic. First, it computes the LAWRR weights of the queues at the beginning of each service round. Next, it obtains a modified weight, which is a product of each queue's priority value and its LAWRR weight. PLAS unconditionally sets the priority value for NRT traffic to 1 but sets that of RT traffic to 2 if the queue size is less than half of the buffer; otherwise, the value is obtained by dividing the number of packets in the RT queue by the LAWRR weight. The algorithm reduces delays in real-time traffic but increases packet drop and reduces average throughput due to its failure to prioritize packets based on their deadlines.

Shareef, Husain, Abdullah & Abdullah, 2002, proposed class-based QoS scheduling (CBS) to increase throughput in existing WiMAX scheduling schemes. The CBS classifies packets into two QoS classes—namely, delay constraint service (DCS) and throughput guarantee service (TGS). The DCS contains the real-time traffics such as UGS and rtPS while the TGS contains nrtPS and BE traffics. The algorithm uses three mechanisms: priority elevation (PE), virtual ranking queue (VRQ), and packet scheduler (PS) to serve packets. First, it prioritizes traffics by setting packet departure rates

to 60% and 40% for weighted high priority (WHP) and weighted low priority (WLP), respectively. Then it groups incoming packets as either WHP or WLP by PE based on their deadlines. Packets with earlier deadlines are grouped as WHP and are moved to VRQ before they are being served by the PS. The algorithm controls packet drops by elevating packets in WLP whose deadlines are about to elapse to VRQ. However, as the number of packets in the VRQ continue to grow under a heavy input traffic, real-time packet transmissions will be interrupted and further delayed; thus, they will increase the packet drop rate of DCS.

Although, the WiMAX standard did not specify any scheduling scheme for its resource management, the scheduling algorithms discussed above were proposed to improve performance of the network even with the limited network resources. Some of these algorithms focus on resource utilization, others on QoS, and still others on fairness. Despite these efforts, there is a need for a scheme that is capable of reducing delay and packet drop and increasing throughput while guaranteeing specific QoS requirements for each service class under varying input traffic rates. Therefore, we propose the EPLAS scheme for resource management in WiMAX networks. The objective of EPLAS is to specifically increase the service rate of queued packets under varying input traffic scenarios for fixed WiMAX in the downlink direction. EPLAS's performance is compared with that of LAWRR, PLAS, and CBS.

EPLAS differs from these schemes, as it employs an adaptive priority weight to reduce delay of real-time traffics by increasing service rate. It also introduces a packet drop control mechanism to address the problem of increased packet drop. Hence, it guarantees QoS of the traffics.

The next section describes the proposed EPLAS scheme.

PROBLEM DEFINITION

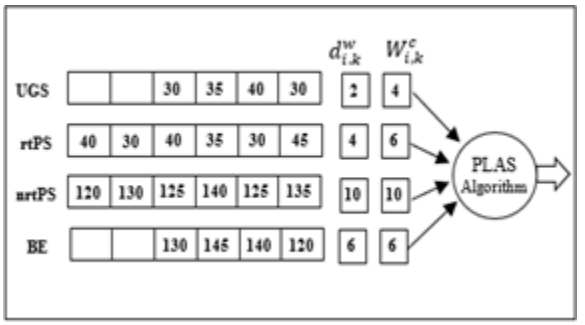
This section presents the weaknesses of PLAS. PLAS is a variant of LAWRR, which separates packets into different queues based on their QoS needs. The algorithm assigns priority weights to each queue at the start of every counter reset. The PLAS priority weight is computed as:

$$P_{i,k}^{w} = P_{i,k}^{v} * d_{i,k}^{w} \tag{1}$$

$P_{i,k}^{v}$ is the priority value of queue i at round k, and $d_{i,k}^{w}$ is the dynamic PLAS weight of queue i at round k. Details on the derivation of Equation (1) are in (Mohammed *et al*., 2018).

Figures 1-4 are used to demonstrate the operation of the PLAS algorithm. Figure 1 shows the state of PLAS after weights are assigned to the weight counter value of each queue. The weight

Figure 1. The state of PLAS after weights are assigned



counter value of a queue determines the number of packets that are served from that queue at the end of every counter reset.

The scheduler serves packets from all non-empty queues in order of their priority as UGS > rtPS > nrtPS > BE. It moves to the next counter reset when all queues are empty or when the weight counter values of all queues are zero. Assuming there is no input traffic during the period of the first counter reset, all packets in all the queues will be served as shown in Figure 2. Therefore, PLAS reduces not only delay and packet drop rate but also increases throughput.

Figure 2 shows that after the first counter reset, the queues become empty and the weight counter values of nrtPS and BE are 4 and 2, respectively. Since the arrival of packets into a queue follows the Poisson distribution (Manirabona et al., 2016), the number of packets in a queue during or after a counter reset does not determine the burstiness of the queue within a specified operation period. Therefore, when there are new arrivals of input traffic into all the queues during the first counter reset of Figure 1, by the time all current packets in the queues are served, the weight counter values of UGS and rtPS will each be zero and those of nrtPS and BE will be 4 and 2, respectively, as shown in Figure 3. This means that the scheduler will skip the UGS and rtPS queues and continue to serve four of the newly arrived packets from nrtPS and two from BE before the next counter reset, as shown in Figure 4. Therefore, the PLAS will cause increased delay of real-time traffic under high-input traffic arrivals. The delay is due to the excess weight allocated to nrtPS and BE during the first counter reset. The standard variance used in PLAS for computing queue load variability led to bias weight values (i.e. it over-weights queues with higher loads and under-weights queues with lower loads) due to the wide difference in the load means of the queues (Meier, 1953; Finch, 2009). Similarly, PLAS will also cause an increase in packet drop as a result of its failure to consider packet deadlines. This

Figure 2. The state of PLAS when all queues are empty

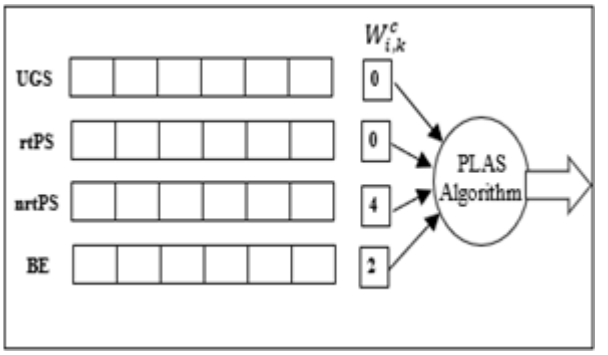


Figure 3. The state of PLAS when new packets arrive before the end of first counter reset

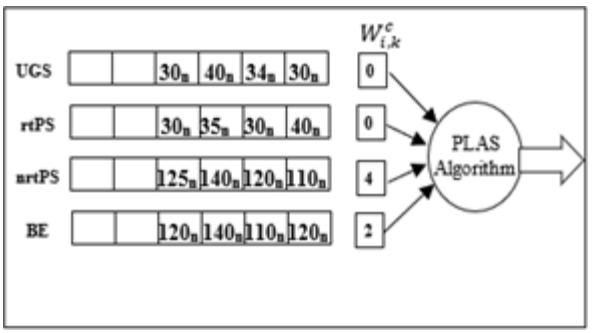
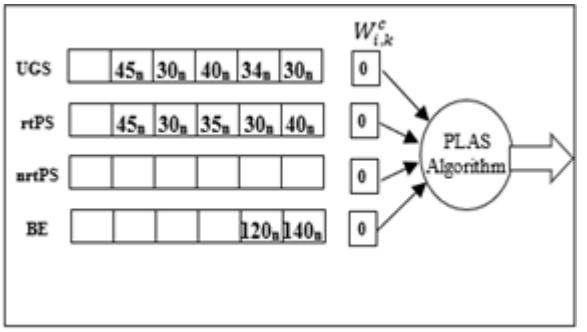


Figure 4. The state of PLAS at the end of first counter reset



is because real-time packets are delay sensitive and the burst times of the packets within each queue vary due to variations in bandwidth allocation during the call admission process (Mohammed, Saidu & Solomon, 2017; Deva et al., 2019). Therefore, under a heavy traffic load, packets with earlier deadlines will be dropped before their service turn.

PROPOSED ALGORITHM

To improve the performance of the existing schemes, the proposed EPLAS addresses the problem of excess weight allocation by employing an adaptive priority weight value. It also employs a packet drop control mechanism to reduce packet drop rate.

Firstly, the adaptive priority weight is derived as follows.

The load weight of queue i at round k is derived as:

$$lw_{i,k} = \frac{k_i}{\sum_{i=1}^{n} k_i} \tag{2}$$

where k_i is the number of packets in queue i and n is the number of queues.

The load of queue i at round k is derived as:

$$l_{i,k} = \sum_{j=1}^{n_i} S_{i,j} \tag{3}$$

where $S_{i,j}$ is the size of the j^{th} packet of queue i.

The weighted load mean of queue i at round k is computed as:

$$G_{i,k} = \frac{1}{\sum_{i=1}^{n_n} lw_{i,k}} * \sum_{i=1}^{n} l_{i,k} * lw_{i,k} \tag{4}$$

where n_n is the number of non-zero load weights.

The load weighted variability of the system at every counter reset is derived as:

$$V_r = \frac{n_n}{n_n - 1} * \sum_{i=1}^{n} \left(l_{i,k} - G_{i,k}\right)^2 * lw_{i,k} \tag{5}$$

where n is the number of queues in the system.

The service weight of queue i at round k is computed as:

$$x_{i,k} = \frac{l_{i,k}}{1 + \sqrt{V_r}} * W_i \tag{6}$$

where W_i is the WRR static weight (Ketevenis *et al.,* 1991).

The EPLAS adaptive priority value is computed as:

$$y_{i,k} = \begin{cases} 2 & if\ UGS\ or\ rtPS\ \&\ \ q_i^n \geq b^f/2 \\ q_i^n / Wt_{i,k} & if\ UGS\ or\ rtPS\ \&\ \ q_i^n < b^f/2 \\ 1 & Other\ classes \end{cases} \tag{7}$$

The priority value is 2 if the queue under consideration is either UGS or rtPS and the number of packets in the queue (q_i^n) is greater than or equal to half of the buffer size (b^f). If the number of packets is less than half of the buffer size, the priority value is the number of packets by the weight of that queue. For nrtPS and BE queues, the priority value is one.

The EPLAS modifies the PLAS priority weight in (Mohammed *et al.,* 2018) by replacing the dynamic weight value $d_{i,k}^w$ in Equation (1) with the new service weight $x_{i,k}$ as follows:

$$Wt_{i,k} = x_{i,k} * y_{i,k} \tag{8}$$

The concept of the weight multiplier value in (Ito, Tasaka & Ishibashi, 2002) was adopted. The authors used it to control the amount of bandwidth allocation among non-empty queues.

Secondly, EPLAS introduces a packet drop control mechanism that prioritizes packets with lower deadlines over others to reduce packet drop rate. The packet drop control mechanism works as follows:

1. All packets that have met their deadlines are dropped from the queue before a packet is served from that queue, and the deadlines of the packets are updated. A packet's deadline is determined by its maximum tolerated latency;
2. A packet that is newly added to a queue is placed ahead of all packets with higher deadlines. The deadline of a packet j of queue i is computed as follows.

The burst time of a packet is derived as:

$$B_{j,i}^t = P_{j,i}^s / BW_{j,i} \tag{9}$$

where $BW_{j,i}$ is the bandwidth allocated to packet j of queue i, and $P^s_{j,i}$ is the packet size of the j^{th} packet of queue i.

The deadline of packet j of queue i is derived as:

$$DL_{j,i} = \left(L_i - {L_i}/{10}\right) - \left[\ B^t_{j,i} + \left(t^c - \ t^a_{j,i}\right)\right] \tag{10}$$

where t^c is the current simulation time, $t^a_{j,i}$ is the arrival time of packet j of queue i, and L_i is the tolerated latency of queue i.

3. Packets from queue *i* are sorted in ascending order based on their deadlines ($DL_{j,i}$).

EPLAS uses weighted variance to compute the load weighted variability of the queues. The choice of this method is intended to produce a more accurate load variance from all the queues despite their differences in load mean. Table 1 and Figure 5 and 6 are used to shows the operation of EPLAS.

The EPLAS weights ($Wt_{i,k}$) computed in Table 1 are assigned to the weight counter ($Wc_{i,k}$) of each queue as shown in Figure 5.

Table 1. Computation of EPLAS modified weight

$q_{i,k}$	k_i	$l_{i,k}$	$x_{i,k}$	$Wt_{i,k}$
$q_{1,1}$	4	0.2	2	4
$q_{2,1}$	6	0.3	4	6
$q_{3,1}$	6	0.3	4	4
$q_{4,1}$	4	0.2	4	4

Figure 5. The state of EPLAS after weights are assigned

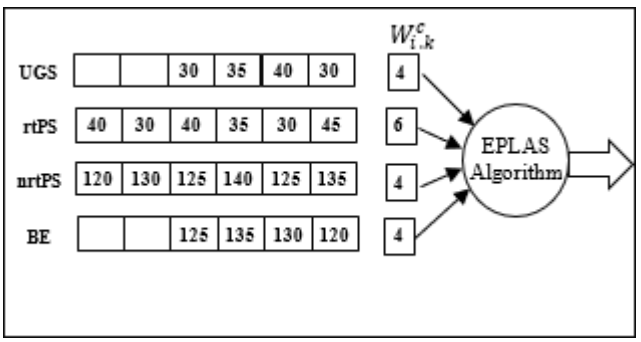


Figure 5 shows the state of EPLAS after weights are assigned. EPLAS, like PLAS, prioritizes real-time traffic by increasing its transmission rate using the adaptive priority value in Equation 5. The weights of UGS and rtPS are increased from 2 and 4 to 4 and 6, respectively. By the end of the first counter reset, the weight counter values of all the queues will be reduced to zero and two packets will be left unserved in the nrtPS queue as shown in Figure 6. The scheduler immediately moves to the next counter reset. The new weight counter values will be computed based on the number of packets in all the queues at the beginning of the counter reset. That is, if no packet arrives during the first counter reset, the weight computation will be based on only the two packets left in nrtPS. But if new packets arrive in all the queues, the scheduler computes the new weight for all the queues and serves the queues starting from UGS to BE. Thus, no packet will be delayed.

The pseudo codes for the EPLAS algorithm are shown in Algorithms 1 and 2.

ALGORITHM 1: EPLAS Packet Drop Control Algorithm

Input:

q_i^n : Number of packets in queue i

b^f : Buffer size

$p_{j,i}$: j^{th} packet of queue i

L_i : Latency of queue i

if $q_i^n \neq$ NULL then

    Compute $DL_{j,i}$ using Equation (10)

  for $j \leftarrow 0$ to $b^f - 1$ do

    if $DL_{j,i} \leq 0$ then

      Drop packet j of queue i

    end if

  end for

    Sort packet in queue i in descending order

    based on $DL_{j,i}$

end if

ALGORITHM 2: EPLAS Scheduling Algorithm

Input:

n : Number of connected queues

$q_{i,k}$: Queue i at round k

$P_{i,k}^{weight}$: Priorityweight of queue i at k

$W_{i,k}^{c}$:Weight counter value of queue i at round k

k : Current RR round

t ←Simulation time

Initializations

$W_{i,k}^{c} = 0$ $(i = 0, 1, 2, \ldots. n - 1)$

$k = 0$

While t IS GREATHER THAN zero do

if $W_{i,k}^{c} = 0$ $(i = 0,1,2, \ldots, n - 1)$ then

drop packets whose deadlines have

elapsed, and sort the queue using the

Algorithm 1

Compute $Wt_{i,k}$ using Equation (8)

$W_{i,k}^{c} = Wt_{i,k}$

end if

for i ←0 to $n - 1$ do

Transmit packet from $q_{i,k}$ using EPLAS

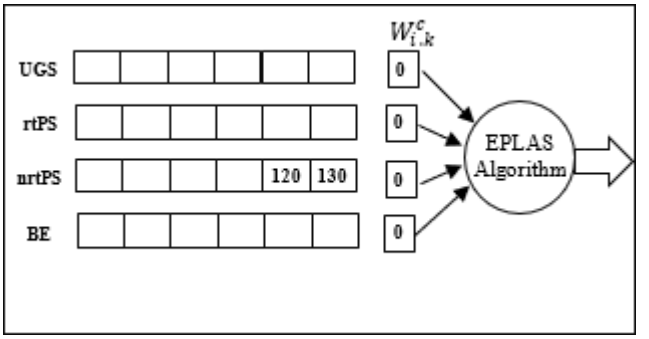
next i

end for

next k

end While

Figure 6. The state of EPLAS after first counter reset



SIMULATION RESULTS

In this section, performance of the proposed EPLAS is compared against the PLAS (Mohammed et al., 2018), CBS (Shareef et al., 2002), and LAWRR (Saidu et al., 2014) scheduling algorithms in terms of delay, packet drop rate, and average throughput. A Java-based discrete event simulator was developed and used for the simulations.

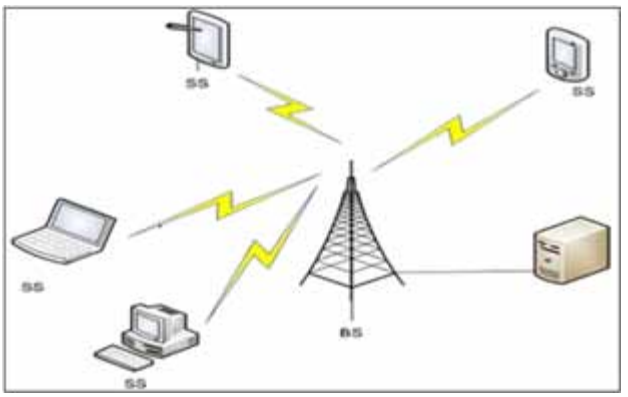
The network topology used was adopted from (Mohammed et al., 2018) as shown in Figure 7. The topology consists of one server, a base station (BS), and a number of SSs connected to the BS for different simulation experiments. The server generates four traffic streams, each from a different application. Each SS carries one traffic stream, and each user uses only one type of traffic at a time. The traffic streams are prioritized based on their QoS requirements as follows: UGS > rtPS > nrtPS > BE.

The simulation experiments were conducted for three input traffic scenarios as follows:

Case 1: 65% of the generated packets are assigned for real-time traffic, and 35% are for non-real-time.
Case 2: 50% of the generated packets are assigned for both real-time and non-real-time traffic.
Case 3: 35% of the generated packets are assigned for real-time traffic, and 65% are for non-real-time.

All simulation experiments used 50, 100, 150, 200, 250, 300, and 350 SSs.

Figure 7. Simulation topology (Mohammed *et al.*, 2018)



Simulation Models

We adopted the traffics models used in (Mohammed et al., 2018) for each QoS class. VoIP traffics are modeled for UGS, video steaming for rtPS, FTP for nrtPS, and HTTP for BE (Table 2).

Performance Metrics

Three metrics were used to measure the performance of the proposed scheduling algorithm against the benchmark algorithms. The metrics are average queuing delay, average packet drop ratio, and average throughput:

- **Average queuing delay:** The time in milliseconds between the arrival and departure of a packet in a queue is expressed as:

$$D_v = \frac{1}{k}\sum_{i=1}^{k}\left[\frac{1}{Nq_i}\sum_{j=1}^{Nq_i}\left(t_j^d - t_j^a\right)\right] \tag{11}$$

where t^a is the packet arrival time, t^d is the packet departure time, Nq_i is the number of packets in queue i, and k is the number of queues in the system.

- **Packet drop ratio:** The ratio of the cumulative number of dropped packets to cumulative queued packets is expressed as:

$$d_r = \frac{p_d}{p_q} \tag{12}$$

Table 2. Simulation parameter (Nie et al., 2011) (Saidu et al., 2014)

Parameters	Values
BS Frequency	2.5 GHz
Duplexing mode	TDD
System bandwidth	5 Mbps
DL/UL ratio	2:1 (29:18 OFDM Symbol)
Frame length	5 ms
Cyclic prefix duration	11.43 μs
Basic symbol	91.43 μs
FFT	1024
PHY	OFDMA
DL permutation	PUSC
MAC PDU length	Variable
Fragment	Enable
ARQ and packing	Disable
DL – UL MAPs	Variable

where p_d is the cumulative packet drop and p_q is the cumulative packet queued.

- **Average throughput:** The number of packets served by the scheduler per unit time in kilobits per seconds is expressed as:

$$T_{av} = \frac{1}{t}\sum_{i}^{n}\left(Nq_i - Mq_i\right) \tag{13}$$

where Nq_i is the number of packets queued in queue i within time t, Mq_i is the number of packets in queue i after time t, t is the total simulation time of the experiment, and n is the number of queues (Tables 3 and 4).

RESULTS AND DISCUSSIONS

Case 1: The goal of this scenario is to subject real-time traffic to a heavier-input traffic load than non-real-time. This is to determine the effectiveness of our modified priority weight and packet drop control scheme against the benchmark algorithms specifically for real-time traffic when the traffic is bursty. Therefore, packet distributions in this scenario for real-time and non-real-time traffic are 65% and 35%, respectively.

Figure 8 shows the average delay incurred by the EPLAS, PLAS, CBS, and LAWRR scheduling algorithms for real-time traffic (UGS and rtPS). The Figure reveals that EPLAS, CBS, and PLAS performed similarly for 50 SSs. This is because of the low traffic generated by the SSs. However, as the traffic increases, the proposed EPLAS shows better performance than PLAS, CBS, and LAWRR.

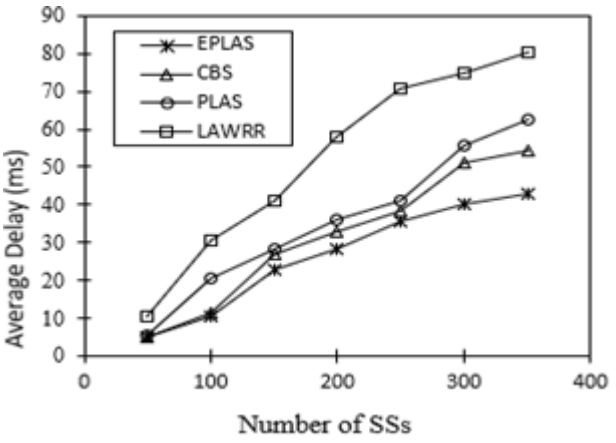
Table 3. Video traffic parameters (Nie *et al.*, 2011) (Saidu *et al.*, 2014)

Parameters	Distribution	Values
ON period	Exponential	Mean 1.34 s
OFF period	Exponential	Mean 1.67 s
Packet size	Constant	66 B
IAT	Constant	20 ms

Table 4. Video streaming parameters (Nie *et al.*, 2011) (Saidu *et al.*, 2014)

Parameters	Value
Video packet size	Geometric (mean= 200B)
Average traffic rate	220 Kbps
MRTR	64 Kbps
MSTR	400 Kbps
Maximum latency	180 ms
Tolerated packet loss	5
IAT	Exponential (mean = 220 Kbps)

Figure 8. Average delay incurred by EPLAS, PLAS, LAWRR, and CBS for real-time (UGS & rtPS) when real-time input traffic is higher than non-real-time



The performance of the proposed algorithms was attributed to the effect of the adaptive priority weight, which reduces excessive weight allocation to non-priority traffic by adaptively adjusting the weight counter values of each queue, thereby reducing the waiting time of the real-time traffic before subsequent counter resets. EPLAS reduces delay incurred in PLAS, CBS, and LAWRR by 25.7%, 15.9%, and 49.5%, respectively.

Figure 9 illustrates the average packet drop ratio by the EPLAS, PLAS, CBS, and LAWRR algorithms for real-time traffics (UGS and rtPS). The Figure shows that the performance of the four algorithms was the same for 50 and 100 SSs, and PLAS, CBS, and EPLAS performed similarly for 150 SSs. However, there was more packet drop in the benchmark algorithms than in EPLAS for 200 to 350 SSs, as EPLAS reduces the packet drop ratio in PLAS by 33.3%, in CBS by 16.7%, and in LAWRR by 57.3%. The EPLAS performance was as a result of the packet drop control mechanism introduced, which prioritizes packets with earlier deadlines within each queue, especially when the traffic is bursty. This mechanism reduces the packet drop rate by forcing the scheduler to serve packets whose deadlines will elapse before they are served.

Figure 9. Average packet drop ratio by EPLAS, PLAS, CBS and LAWRR for real-time (UGS & rtPS) when real-time input traffic is higher than non-real-time

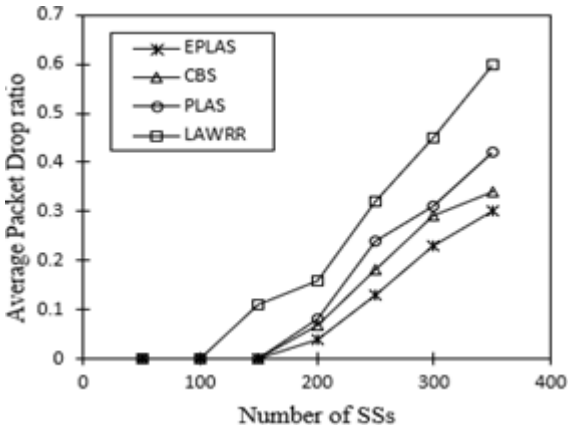


Figure 10 shows the average throughput from real-time traffic (UGS and rtPS) by the EPLAS, CBS, PLAS, and LAWRR algorithms. This Figure demonstrates that the four algorithms performed similarly for 50 and 100 SSs. The benchmark algorithms also had similar performances for 150 SSs, but EPLAS performed better for 150 to 350 SSs. The performance of EPLAS is due to the decrease in delay and packet drop and the higher priority given to real-time traffic, which causes more packets to be transmitted. EPLAS improved the throughput in PLAS, CBS, and LAWRR by 18.7%, 11.7%, and 29.1%, respectively.

Case 2: The objective of this scenario is to study the performance of the four algorithms when real-time traffic and non-real-time traffic are equally bursty. We assigned 50% of the generated packets to each of the traffic classes.

Figure 11 shows the average delay incurred by the EPLAS, PLAS, CBS, and LAWRR algorithms. This Figure reveals that EPLAS outperforms the benchmark algorithms for all the SSs.

Figure 10. Average throughput for real-time traffic (UGS and rtPS) by EPLAS, PLAS, CBS and LAWRR algorithms for different SSs when real-time input traffic is higher than non-real-time

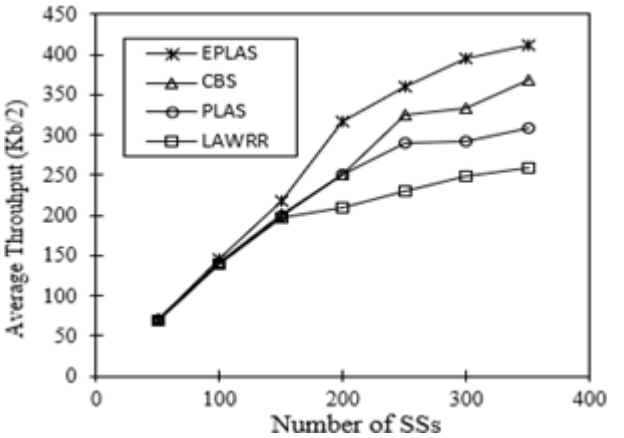
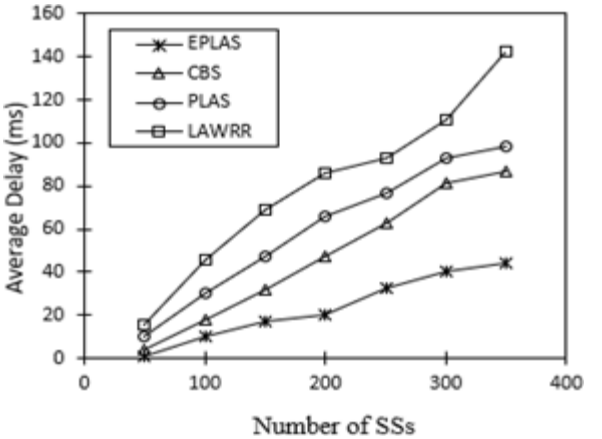


Figure 11. Average delay incurred by EPLAS, PLAS, CBS and LAWRR for real-time (UGS & rtPS) classes for different SSs when real-time traffic and non-real-time traffic are equally bursty



As in Case 1, the performance is due to the weight adjustment of the queues, which control the excessive weight allocation to non-real-time traffic by the benchmark algorithms. The benchmark schemes incurred more delay because of the LAWRR dynamic coefficient that is common in computation of their weight values. EPLAS reduced the delay in PLAS, CBS, and LAWRR by 60.9%, 50.3%, and 70.6%, respectively.

Figure 12 shows the average packet drop by the EPLAS, PLAS, CBS, and LAWRR algorithms for real-time traffic (UGS and rtPS). It can be observed that a similar performance was demonstrated by the four algorithms for 50 and 100 SSs. For 150 SSs, the performances of PLAS and LAWRR and CBS and EPLAS were also similar. The proposed scheme shows superiority compared to PLAS, CBS, and LAWRR as the traffic rate increases (200 to 350 SSs). The proposed algorithm performed better because the increase in traffic in non-real-time queues increases delay in the benchmark schemes, which consequently increases packet drop as a result of the failure of PLAS, CBS, and LAWRR to consider packet deadlines within the queues. EPLAS was able to reduce the packet drop ratio of PLAS by 43.7%, CBS by 26%, and LAWRR by 63.2%.

Figure 13 demonstrates the average throughput by the EPLAS, PLAS, CBS, and LAWRR algorithms for real-time traffic. The Figure shows very little difference in the performance of the

Figure 12. Average packet drop ratio by EPLAS, PLAS, CBS and LAWRR for real-time (UGS & rtPS) classes for different SSs when real-time traffic and non-real-time traffic are equally bursty

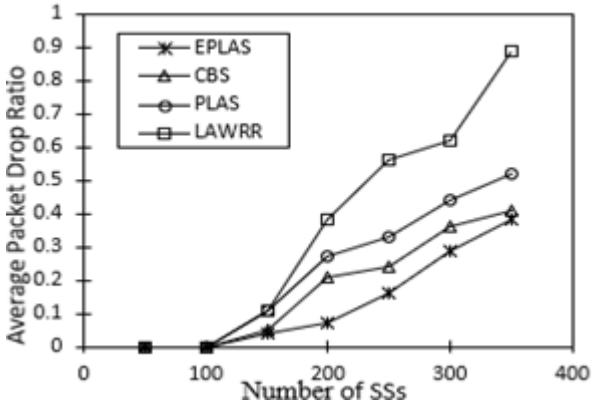
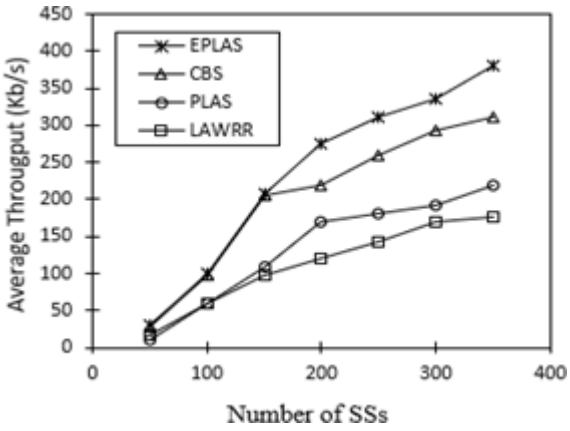


Figure 13. Average throughput for real-time traffic (UGS & rtPS) by EPLAS, PLAS, CBS and LAWRR algorithms for different SSs when real-time input traffic is higher than non-real-time



four algorithms for 50 SSs. EPLAS performed similarly as CBS for 50 to 150 SSs. This is because of their similar performance in terms of packet drop. However, the proposed algorithm demonstrated superior performance compared to the benchmark algorithms for 200 to 350 SSs, as it increased the throughput in PLAS by 42.7%, in CBS by 16.1%, and in LAWRR by 52.1%. This is because when the traffic became less bursty, there were fewer instances of buffer overflow, which usually causes packet drop. Also, the prioritization of packets with earlier deadlines and the introduction of the modified priority weight caused a decrease in packet drop and consequently increased the throughput.

Case 3: In this scenario, we generated 35% of the packets for real-time queues and 65% for non-real-time queues.

Figure 14 demonstrates the performance of EPLAS, PLAS, CBS, and LAWRR in terms of average packet delay in real-time traffics (UGS and rtPS). It can be observed that the four algorithms performed similarly for 50 and 100 SSs. CBS performed slightly better than EPLAS for 200 to 250 SSs. The proposed algorithm shows better performance compared to the benchmark algorithms for 200 to 350 SSs. The better performance was because, apart from the introduction of the modified priority weight, which increases the service rate of real-time traffic, the low-input traffic of real-time queues reduced the waiting time of packets in the queues. Therefore, EPLAS reduced delay in PLAS by 22.2%, in CBS by 9.12%, and in LAWRR by 26.2%.

Figure 15 illustrates the average packet drop ratio by the EPLAS, PLAS, CBS, and LAWRR algorithms for real-time traffic. This Figure shows that the four algorithms performed similarly for 50 to 150 SSs. However, when the number of SSs increased and the traffic intensity also increased, the proposed algorithm performed significantly better by reducing the packet drop in PLAS by 25%, in CBS by 2.9%, and in LAWRR by 34%.

This is because when the input traffic is low, the proposed algorithm adaptively modifies the weight counter values of queues to serve 80-100% of the packets with earlier deadlines in the queue.

Figure 16 shows the average throughput by the EPLAS, PLAS, CBS, and LAWRR algorithms. This Figure reveals that the performance of the four algorithms was the same for 50 to 150 SSs due to very low traffic generated by the SSs. Though EPLAS showed similar performance to PLAS and CBS for 200 SSs, it increased the average throughput of PLAS, CBS, and LAWRR by 4.6%, 0.7%, and 12.6%, respectively. This is because when the traffic in real-time queues was low

Figure 14. Average delay incurred by four algorithms for real-time (UGS & rtPS) classes for different SSs when real-time traffic is less bursty than non-real-time traffic

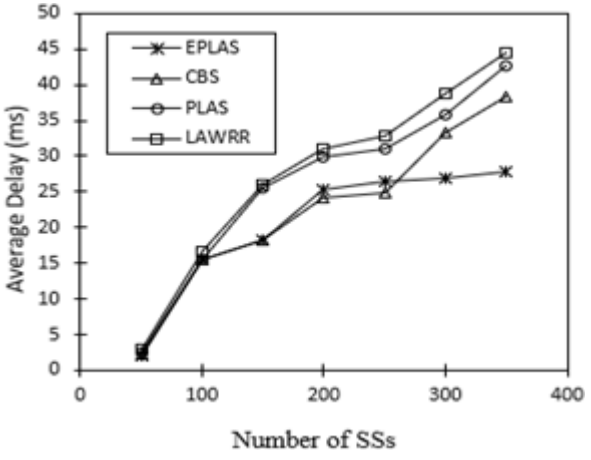


Figure 15. Average packet drop ratio by EPLAS, PLAS, CBS, and LAWRR for real-time (UGS & rtPS) classes for different SSs when real-time traffic is less bursty than non-real-time traffic

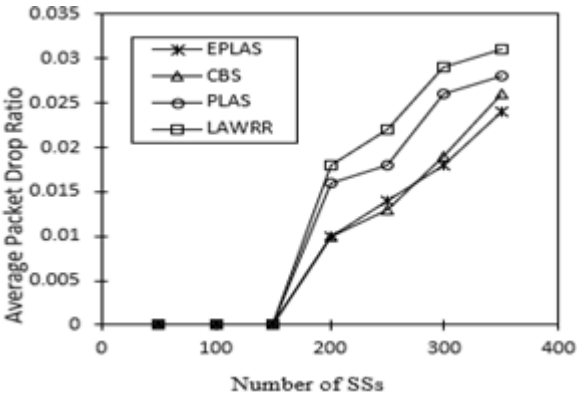
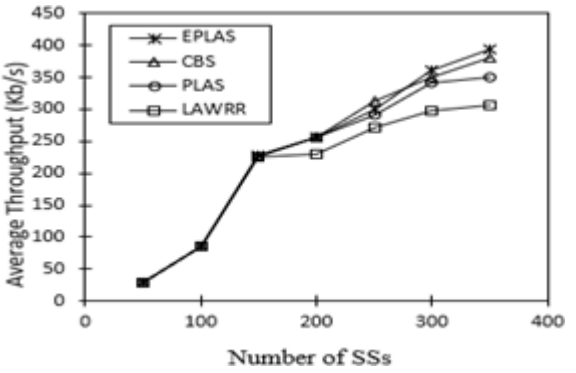


Figure 16. Average throughput for real-time traffic (UGS & rtPS) by EPLAS, PLAS, CBS and LAWRR algorithms for different SSs when real-time traffic is less bursty than non-real-time traffic



compared to non-real-time, EPLAS modified the priority weight and adaptively adjusted queue weights to avoid allocation of excess weight to non-real-time traffic. This greatly reduced the waiting time of real-time packets compared to the benchmark schemes. Thus, EPLAS reduces packet drop and increases throughput.

CONCLUSION AND FUTURE WORK

In this paper, we proposed an enhanced priority load-aware scheduling for wireless broadband networks to improve the performance of existing scheduling algorithms in WiMAX. The proposed scheme employs a priority weight that adaptively adjusts the weight counter value of each queue based on the load and number of packets in the queue. The weight prioritizes real-time traffics to reduce delay. EPLAS also introduces a packet drop control mechanism to reduce the packet drop rate in real-time traffics. A discrete event simulator was designed and used to evaluate the performance of the proposed scheme against the benchmark algorithms. Simulation experiments were conducted for three traffic scenarios for a varying number of SSs. In the first scenario, the traffic distribution of real-time and non-real-time traffics was 65% and 35%, respectively. In the second scenario, real-time and non-real-time traffics each had 50% of the traffics. In the third scenario, the distribution

of real-time and non-real-time traffics was 35% and 65%, respectively. The results of the simulation show that the proposed algorithm achieves better performance compared to other schemes as it was able to reduce average delay and packet drop and increase average throughput of real-time traffics under different network sizes. The simulation experiments were validated using an analytical model, and the results match with insignificant difference. Therefore, not only does EPLAS achieve better resource utilization, it also guarantees QoS of real-time traffics.

This work considered fixed WiMAX. Therefore, in the future, EPLAS can be enhanced by integrating a mechanism that handles channel quality variation to support mobile WiMAX and other broadband wireless technologies such as long-term evolution (LTE).

REFERENCES

Ahmed, Z., Hamma, S., & Nasir, Z. (2019). An optimal bandwidth allocation algorithm for improving QoS in WiMAX. *Multimedia Tools and Applications*, 1–40.

Avocanh, F. T. S., Abdennebi, M., & Ben-Othman, J. (2014). An Enhanced Two-Level Scheduler to Increase Multimedia Services Performance in LTE Networks. In *Proceedings of the IEEE International Conference on Communications (ICC)* (pp. 2351-2356). IEEE Press. doi:10.1109/ICC.2014.6883674

Chin-Ling, C., & Cheng-Yi, P. (2012). A Downlink Scheduling Based on Queue Length Estimation for VoIP in WiMAX Networks. In *Proceedings of the IEEE 15th International Conference on BioMedical Engineering and Informatics*. IEEE Press.

Comsa, I.-S., De-Domenico, A., & Ktenas, D. (2019). U.S. Patent No. US 2019 / 0124667 A1. Washington, DC: U.S. Patent and Trademark Office.

Comsa, I.-S., Trestian, R., Muntean, G.M., and Ghinea, G. (2020). 5MART: A 5G SMART Scheduling Framework for Optimizing QoS Through Reinforcement Learning. *IEEE Transactions on Network and Service Management*.

Comsa, I.-S., Zhang, S., Aydin, M., Kuonen, P., Trestian, R., & Ghinea, G. (2019b). Enhancing User Fairness in OFDMA Radio Access Networks Through Machine Learning. In Wireless Days (WD) (pp. 1-8). Academic Press. doi:10.1109/WD.2019.8734262

Comsa, I.-S., Zhang, S., Aydin, M. E., Kuonen, P., Lu, Y., Trestian, R., & Ghinea, G. (2018). Towards 5G: A Reinforcement Learning-Based Scheduling Solution for Data Traffic Management. *IEEE eTransactions on Network and Service Management*, *15*(4), 1661–1675. doi:10.1109/TNSM.2018.2863563

Comsa, I.-S., Zhang, S., Aydin, M. E., Kuonen, P., Trestian, R., & Ghinea, G. (2019a). Guaranteeing User Rates with Reinforcement Learning in 5G Radio Access Networks. In I. S. Comsa & R. Trestian (Eds.), *Next-Generation Wireless Networks Meet Advanced Machine Learning Applications* (pp. 163–198). Hershey, PA: IGI Global. doi:10.4018/978-1-5225-7458-3.ch008

Deva, P. M., Sangeetha, M., Christy, J. M. A., Dhivyaprabha, E., Kiruthiga, N., & Rajarajesuwari, P. L. (2019). Fair Adaptive Cross-Layer Resource Allocation Scheme for IEEE 802.16 Broadband Wireless Networks. *Wireless Personal Communications*, *109*(1), 1–22.

Finch, T. (2009). *Incremental Calculation of Weighted Mean and Variance.* University of Cambridge Computing Service.

Hahne, E. L. (1991). Round-Robin Scheduling for Max-Min Fairness in Data Networks. *IEEE Journal on Selected Areas in Communications*, *9*(7), 1024–1039. doi:10.1109/49.103550

Iaad, B. D., Mustapha, B., Taoufik, E., Samer, L., & Xavier, L. (2019). Dynamic Access Point Selection and Resource Allocation in Multi-Technology Wireless Network. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE Press.

Ito, Y., Tasaka, S., & Ishibashi, Y. (2002). Variably Weighted Round Robin Queueing for Core IP Routers. In *Proceedings of the 21st IEEE International Conference on Performance, Computing, and Communications* (pp. 159–166). IEEE Press. doi:10.1109/IPCCC.2002.995147

Ketevenis, M., Sidiroipoulos, S., & Courcoubetis, C. (1991). Weighted Round Robin Cell Multiplexing in a General-Purpose ATM Switch Chip. *IEEE Journal on Selected Areas in Communications*, *9*(8), 1265–1279. doi:10.1109/49.105173

Khan, N., Martini, M. G., Bharucha, Z., & Auer, G. (2012). Opportunistic Packet Loss Fair Scheduling for Delay-Sensitive Applications over LTE Systems. In *Proceedings of IEEE Wireless Communications and Networking Conference* (pp. 1456-1461). IEEE Press. doi:10.1109/WCNC.2012.6214010

Manirabona, A., Boudjit, S., & Fourati, L. C. (2016). *A Priority- Weighted Round Robin Scheduling Strategy for a WBAN Based Healthcare Monitoring System.* In *Proceedings of the IEEE 13th Annual Consumer Communications & Network Conference* (pp. 224–229). IEEE Press.

Mardini, W., & Alfool, M. A. (2011). Modified WRR Scheduling Algorithm for WiMAX Networks. *Network Protocols and Algorithms Journal*, *3*(2), 24–53.

Meier, P. (1953). Variance of Weighted Mean. *International Biometrics Society*, *9*(1), 59–73. doi:10.2307/3001633

Mohammed, A., Saidu, I., & Abdulazeez, A. (2018). A Priority Load-Aware Scheduling (PLAS) Algorithm for Wireless Broadband Networks. In V. Odumuyiwa, O. Adegboyega, & C. Uwadia (Eds.), e-Infrastructure and e-Services for Developing Countries (pp. 49–59). Springer, Cham, 250, .

Mohammed, A., Saidu, S., Isah, B. Y., & Solomon, Y. (2017). A Dynamic QoS-Aware Call Admission Control Algorithm for Mobile Broadband Networks. In *Proceedings of the IEEE International Conference on Computing, Networking, and Informatics (ICCNI)*. IEEE Press. doi:10.1109/ICCNI.2017.8123772

Naik, D., Dora, S., & De, T. (2019). Normalized Uplink Bandwidth Scheduling Algorithm for WiMAX Networks. In Advances in Computer, Communication and Control (pp. 311-325). Springer.

Nie, W., Wang, H., & Park, J. H. H. (2011). Packet Scheduling with QoS and Fairness for Downlink Traffic in WiMAX Networks. *Journal of Information Processing Systems*, *7*(2), 261–270. doi:10.3745/JIPS.2011.7.2.261

Rajeem, P. & Fernando, J. J. V. (2010). OFDMA WiMAX Physical Layer. In WiMAX Networks (pp. 63–135). Springer.

Saidu, I., Subramaniam, S., Jaafar, A., & Zukarnain, Z. A. (2014). A Load-Aware Weighted Round-Robin Algorithm for IEEE 802.16 Networks. *EURASIP Journal on Wireless Communications and Networking*, *13*(18), 226. doi:10.1186/1687-1499-2014-226

Shareef, Z. A., Hussin, M., Abdullah, A., & Abdullah, M. (2018). Class-based QoS Scheduling of WiMAX networks. *Journal of High-Speed Networks*, *24*(4), 345–362. doi:10.3233/JHS-180599

Wu, S. I., Huang, S. Y., & Huang, K. F. (2012). Adaptive Priority-Based Downlink Scheduling for WiMAX Networks. *IEEE Trans. Journal of Communications and Networks (Seoul)*, *14*(6), 692–702. doi:10.1109/JCN.2012.00035

Zou, L., Trestian, R., & Muntean, G.-M. (2013a). A utility-based priority scheduling scheme for multimedia delivery over LTE networks. In *Proceedings of the 2013 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)* (pp. 1–7). IEEE Press. doi:10.1109/BMSB.2013.6621745

Zou, L., Trestian, R., & Muntean, G.-M. (2013b). Device-oriented energy-aware utility-based priority scheduler for video streaming over LTE System. In *Proceedings of the Information Technology and Telecommunications Conference (IT&T)*. Academic Press.