A Priority Load-Aware Scheduling Algorithm for Wireless Broadband Networks

Aminu Mohammed[1], Ibrahim Saidu[2], and Abdulhakeem Abdulazeez[1](✉)

[1] Department of Mathematics, Computer Science Unit, Usmanu Danfodiyo University, P.M.B 2346, Sokoto, Nigeria
{mohammed.aminu, abdulhakeem.abdulazeez}@udusok.edu.ng
[2] Department of Information and Communications Technology, Usmanu Danfodiyo University, P.M.B 2346, Sokoto, Nigeria
ibrahim.saidu@udusok.edu.ng

Abstract. Wireless broadband networks are emerging as reliable internet access alternatives for delivery of high speed multimedia services. WiMAX is one of such networks, designed to provide quality of service (QoS) support for different service classes with varying QoS requirements. Scheduling algorithms are required to provide such support. The existing scheduling algorithm uses dynamic weight to allocate resources based on traffic loads. However, it increases delay of real time traffics due to failure of the weight to prioritize traffics. This paper proposes a priority load aware scheduling (PLAS) algorithm to reduce delay in real time traffics. The PLAS algorithm introduces a priority value to prioritize real time traffics over non-real time traffics. The algorithm was evaluated using extensive simulations. The results show that the PLAS outperforms the existing algorithm in terms of delay.

Keywords: WRR · Scheduling algorithm · WiMAX · QoS

1 Introduction

Wireless broadband networks have become reliable means to meet the increasing demand for internet connections and integrated multimedia services. WiMAX being one of such networks provides last mile internet access to both residential and enterprise users. It defines the physical (PHY) layer and media access control (MAC) layer. Apart from its ease and cheap cost of deployment as well as maintenance, it also supports multiple QoS classes [1]. The QoS classes are as follows:

I. Unsolicited Grant Service (UGS) is designed for real-time services like Voice over Internet Protocol (VoIP)) that require constant bit rate flows. UGS is given higher priority because of its low tolerant for delay and requirement for maximum latency.
II. real time Polling Service (rtPS) is designed for real-time services such as MPEG that generates variable data size periodically. It is more delay tolerant than the UGS.

III. non-real time Polling Service (nrtPS) is designed for non real-time services like file transfer protocol (FTP) which also generates variable data size periodically. This class is delay tolerant and has minimum bandwidth requirement.
IV. Best Effort (BE) is intended for services that do not require QoS guarantee such as Hyper Text Transfer Protocol (HTTP).

The MAC layer is responsible for scheduling these classes with varying QoS requirements such as bandwidth, delay, jitter and packet loss. To satisfy these requirements, an efficient scheduling algorithm is needed.

Several scheduling algorithms have been proposed for resource allocation [2–9]. Recently, Load Aware Weighted Round Robin (LAWRR) has been proposed to mitigate the weakness of Weighted Round Robin (WRR). The LAWRR separates packets into different QoS classes and allocates a dynamic weight to each queue based on its traffic load characteristics. The algorithm employs the dynamic weight to determine the number of packets to be served in each queue. It reduces not only delay and packet loss but also improves average throughput for non-real time traffics. However, it increases delay due to its failure to prioritize real time traffics.

In this paper, a priority load aware scheduling (PLAS) algorithm is proposed to reduce delay in real time traffics. The PLAS introduces a priority weight to increase the service rate of the real time traffics. The algorithm is evaluated using simulations. The result demonstrates that the PLAS achieves superior performance compared to the existing scheme.

The rest of this paper is organized as follows: Sect. 2 presents related works. In Sect. 3, the proposed PLAS is prescribed; Sect. 4 presents performance evaluation while Sect. 5 concludes the paper.

2 Related Works

In this section, some of the scheduling algorithms proposed for WiMAX networks are reviewed as follows:

[1], proposed a priority weighted round robin (PWRR) scheduling algorithm to minimize delay. The PWRR employs a classifier that separate packets based on their priorities. The algorithm employs the priority scheduling (PS) and the weighted round robin (WRR) disciplines to determine packets to transmit. The PS schedule packets from the high priority classes while WRR from low priority classes. The PWRR reduces delay and increases throughput of high priority traffics but starves the low priority classes.

In [9], a modified weighted round-robin (MWRR) scheduling algorithm is proposed to avoid starvation of lower priority classes. The MWRR algorithm assigns weight to each active queue based on its priority. The algorithm multiplies the static WRR and a constant multiplier value to increase the number of packets to be serviced from each queue. The multiplier is an integer value obtained base on a network size; the larger the network the smaller the multiplier value and vice versa. The MWRR reduces delay and increases throughput but may lead to an increase in average delay and decrease in average throughput if the multiplier is wrongly selected.

In [4], a variant of WRR known as adaptive weighted round-robin (AWRR) scheduling algorithm is also proposed to reduce starvation of lower service classes. The AWRR uses input and output schedulers to adjust service classes' weights. The input scheduler gives priorities to high priority traffic over low priority traffic based on their bandwidth and latency requirements. On the other hand, the output scheduler controls data flows and manage the service classes. The algorithm also employs a threshold for each class which when exceeded triggers dynamic weight adjustment. It reduces delay as well as improves throughput. However, the algorithm increases average delay and packet loss as well as decrease in throughput if the threshold value is inappropriately set.

[5], proposed a low latency weighted round robin (LLWRR) scheduling algorithm to improve latency and fairness in high priority traffics. The LLWRR computes varying weights by multiplying a coefficient value and the static WRR weight at the beginning of each service round. The coefficient value is a function of number of queues, which decreases as the number of queues increases. LLWRR adjusts the weight values based on number of queues. The algorithm improves not only fairness and average throughput but also decreases latency. However, it may lead to an increase in average delay and packet lost under busty input traffics with large number of queues.

[3], proposed a load-aware weighted round-robin (LAWRR) algorithm to address the WRR performance degradation problem under busty traffic condition. The LAWRR algorithm dynamically computes weight of a queue according to its current loads characteristics. The weight of each queue is computed as the product of the dynamic coefficient of a queue and the static WRR weight at the beginning of each base station (BS) round. The dynamic coefficient is obtained by computing the root mean square error from the load variance of the queues. The algorithm reduces not only average delay and packet loss but also improves average throughput for non real- time traffics. However, it increases delay due to starvation of real time traffics.

In addition to the scheduling algorithms reviewed above, other resource management schemes are proposed in [10, 11]. However, to the best of our knowledge, none of the existing algorithms is capable of decreasing delay of real time traffics by prioritization using dynamic weight.

3 The Proposed PLAS Algorithm

This section describes the proposed scheduling algorithm named PLAS. The PLAS is a variant of the LAWRR. However, first, the weakness of LAWRR is described. The LAWRR separates packets into different service classes considering their QoS requirements. It uses a dynamic weight to service each class. The weight is computed in Eqs. (1)–(4) as follows:

First, the root mean square error is computed as:

$$R_{r=\sqrt{v_r}}. \tag{1}$$

Where v_r is the load variance of the queues.

Then the dynamic coefficient of queue i at round r is derived as:

$$w_{i,r} = \left\lceil \frac{L_{i,r}}{R_r + 1} \right\rceil. \tag{2}$$

Where $L_{i,r}$ is the load of queue i at round r.

Next, the static WRR weight is computed as:

$$Sw_i = \frac{MRTR_i}{\sum_{i=1}^{n} \mathrm{MRTR_i}}. \tag{3}$$

Where $MRTR_i$ is the minimum reserve traffic rate of queue i and n is the total number of queues in a class.

Finally, a dynamic weight of queue i at round r is obtained as:

$$dw_{i,r} = w_{i,r} Sw_i. \tag{4}$$

The algorithm employs Eq. (4) to determine number of packets to be served in each queue. It assigns a weight counter to all non-empty queues. At the beginning of every counter reset, a dynamic weight is assigned to each weight counter. If a packet is sent from a queue, the weight counter value of that queue is decremented by 1. Every queue is served once in every round in a round robin (RR) fashion. The scheduler moves to next round when all queues are served in the current round. The scheduler continues in this fashion until the weight counter of each queue is zero or all queues are empty. To demonstrate the algorithm, Fig. 1 is used.

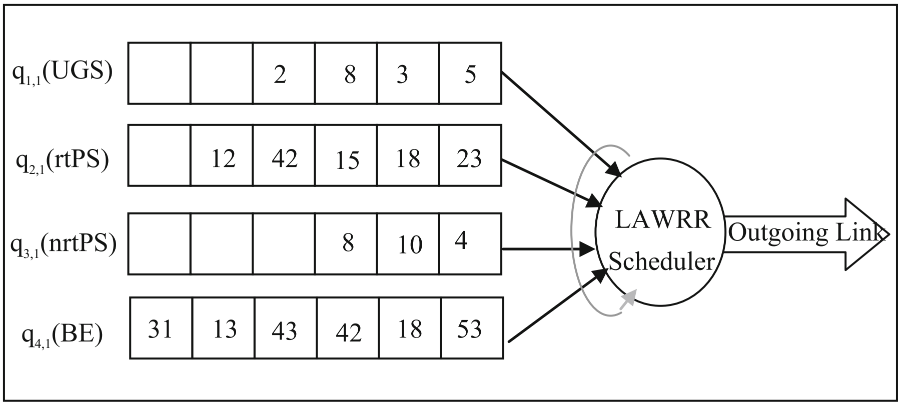

Fig. 1. Shows the state of a LAWRR scheduler before weights assignment.

From Fig. 1, the dynamic weights are computed using Eq. (4) and shown in Table 1. The dynamic weights are assigned to the weight counters of the queues shown in Fig. 2. The queues in this Fig. are served starting from $q_{1,r}$ to $q_{4,r}$ in every service round and the counter weight of each queue is decremented by 1 when a packet is served. By the end of second round in first counter reset, the counter values of $q_{1,2}$ and

$q_{3,2}$ become zero each while that of $q_{2,2}$ and $q_{4,2}$ become 2 and 4, respectively as shown in Fig. 3. That means, no queue will be served from $q_{1,2}$ and $q_{3,2}$ until after the next four rounds when all counter weights become zero and the counter values are reset. By the end of first counter reset, a total of 3 packets remain in $q_{1,6}$ (UGS) and $q_{2,6}$(rtPS) as shown in Fig. 4. This example shows that packets waiting in UGS and rtPS classes are delayed at the end of every counter reset until subsequent counter reset. However, the delay incurred in these queues may cause packet drop and decrease in average throughput in real time traffics. Therefore, LAWRR is only suitable for nrtPS and BE but not for real time traffics.

Table 1. Computation of dynamic weight for LAWRR

$q_{i,1}$	$L_{i,1}$	$L_{i,1} - L_1$	$dw_{i,1}$
$q_{1,1}$	18	4,830.25	2
$q_{2,1}$	110	506.25	4
$q_{3,1}$	22	4,290.25	2
$q_{4,1}$	200	12,656.25	6
Total	**350**	**22,283**	
Stat.	**L1 = 87.5**	**v1 = 5,570.75, R1 = 74.6375**	

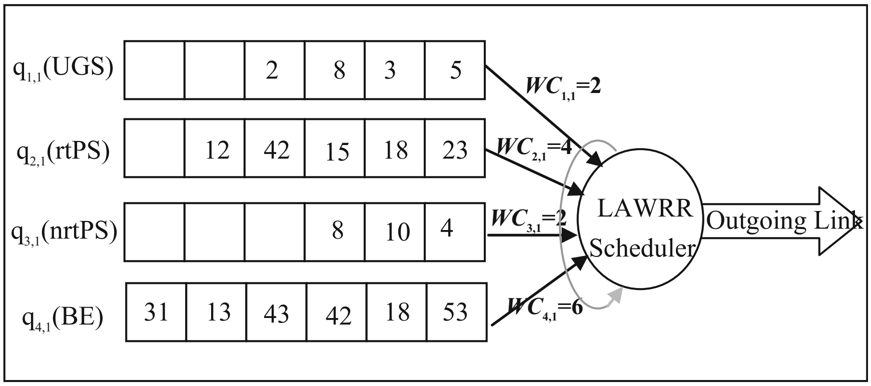

Fig. 2. Shows the state of a LAWRR after weights assignment

To address the shortcomings of the LAWRR, a PLAS algorithm is proposed to increase the service rate of real time traffics. First, the scheme modifies the LAWRR weight in Eq. (4) by introducing a priority value derived as follows:

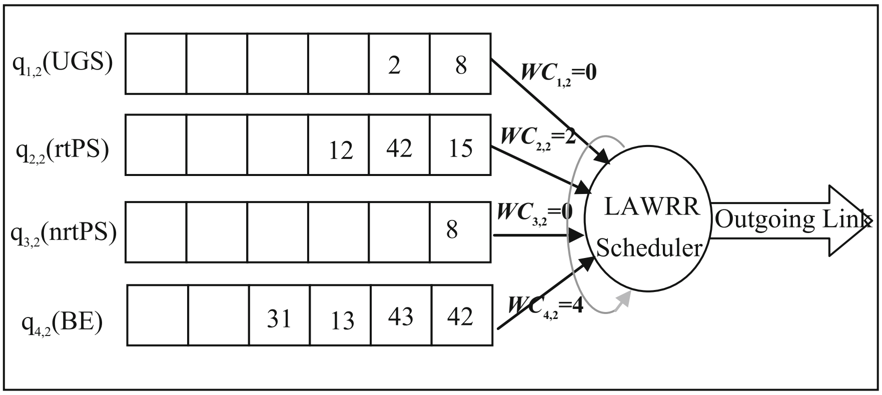

Fig. 3. Shows the LAWRR after second round.

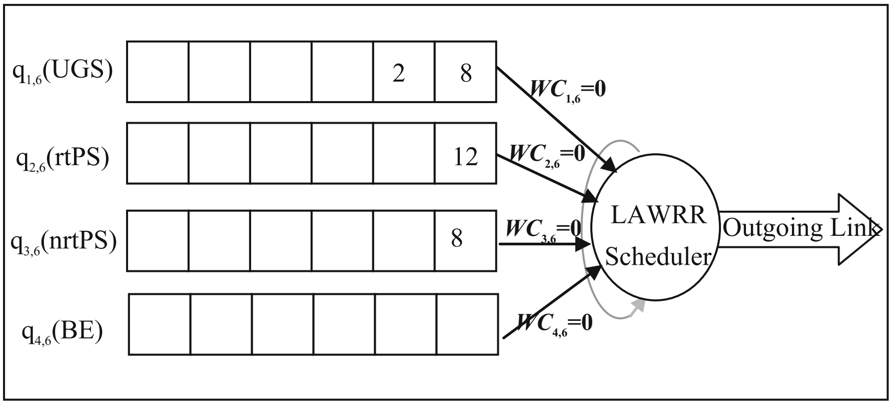

Fig. 4. Shows the LAWRR after last round

$$Pv_{i,r} = \begin{cases} 2 & \textit{if UGS or rtPS \&} \quad Nq_i \leq \quad Bf/2 \\ Nq_i/dw_{i,r} & \textit{if UGS or rtPS \&} \quad Nq_i > \quad Bf/2 \\ 1 & \textit{Otherwise} \quad . \quad . \end{cases} \tag{5}$$

Where $dw_{i,r}$ is the LAWRR dynamic weight of queue i at round r, Bf is the size of buffer and Nq_i is the number of packets in queue i.

Then, the modified weight of queue i at round r is computed as:

$$Mw_{i,r} = Pv_{i,r} \,.\, dw_{i,r}. \tag{6}$$

Also, Fig. 1 is used to demonstrate the effect of the proposed PLAS. The Fig. and Eq. (6) are used to compute the modified weights as shown in Table 2. The table shows that the modified weights are computed and represented as: $Mw_{1,1}$ = 4, $Mw_{2,1}$ = 4,

$Mw_{3,1}$ = 2, and $Mw_{4,1}$ = 6. These weights are assigned to respective weight counters as: $WC_{1,1}$ = 4, $WC_{2,1}$ = 4, $WC_{3,1}$ = 2, and $WC_{4,1}$ = 6 as shown in Fig. 5. This Fig. schedules packet as:

$q_{1,1} \rightarrow q_{2,1} \rightarrow q_{3,1} \rightarrow q_{4,1} \rightarrow q_{1,2} \rightarrow q_{2,2} \rightarrow q_{3,2} \rightarrow q_{4,2} \rightarrow q_{1,3} \rightarrow q_{2,3} \rightarrow q_{4,3} \rightarrow q_{1,4} \rightarrow q_{2,5} \rightarrow q_{4,4} \rightarrow q_{2,5} \rightarrow q_{4,5} \rightarrow q_{4,6}$

Table 2. Modified weight computation

$q_{i,1}$	$Nq_{i,1}$	$dw_{i,1}$	$Pv_{i,1}$	$Mw_{i,1}$
$q_{1,1}$	4	2	2	4
$q_{2,1}$	5	4	5/4	5
$q_{3,1}$	3	2	1	2
$q_{4,1}$	6	6	1	6

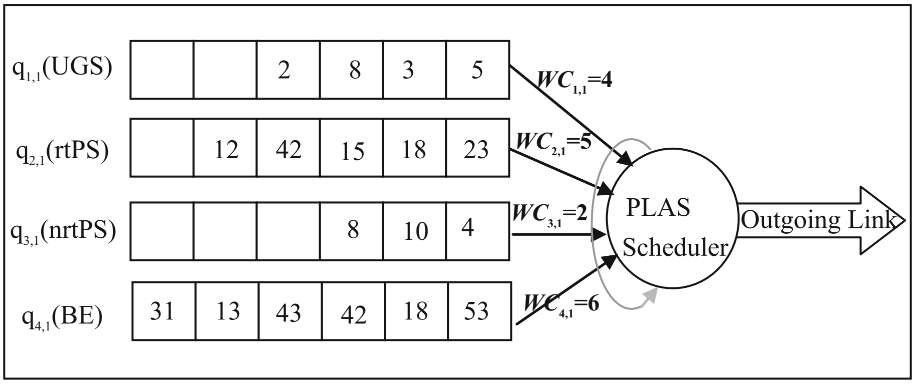

Fig. 5. Shows the state of a PLAS after weights are assigned

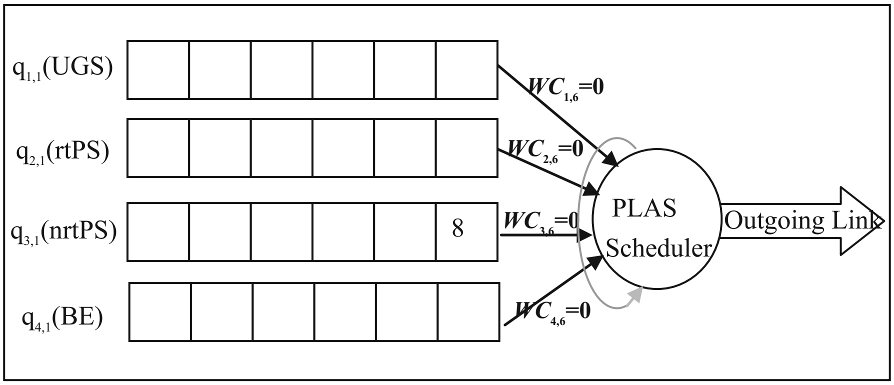

Fig. 6. Shows the PLAS after weights have been exhausted

After packets are scheduled from Fig. 5, only one packet is queued before the next counter reset as shown in Fig. 6. In this Fig., as compared with the LAWRR scheduler in Fig. 4, the PLAS scheduler reduced the total number of delayed packets from four to one. Hence, the number of delay sensitive packets ($q_{1,6}$ and $q_{2,6}$) that were delayed have been reduced from three to zero.

The Pseudo Code for the algorithm is shown in Algorithm 1.

Algorithm 1. PLAS Algorithm

```
1   n ← number of connected queues
2   $q_{i,r}$ ← queue I at round r
3   $Mw_{i,r}$ ← modified weight of queue i at round r
4   $WC_{i,r}$ ← weight counter value of queue I at round r
5   r ← current RR round
6   $WC_{i,r}$ ← $0(i=1,2...n-1)$
7   r ← 0
8   for $i$ ←0 to $n-1$ do
9     if $q_{i,r} \neq$ NULL, then
10        Compute $Mw_{i,r}$ using Equation 6
11        $WC_{i,r}$ ← $Mw_{i,r}$
12        if $q_{i,r} \neq$NULL and $WC_{i,r} \neq$ NULL then
13          Transmit packet from $q_{i,r}$ using WRR
14        else
15          r ← r +1
16    else
17      Next $i$
18  end
```

4 Performance Evaluation

In this section, we compare the performance of LAWRR [3] with the proposed PLAS scheduling algorithm in terms of average delay. A discrete-event simulator was developed and used to conduct simulations for the evaluation. The simulation parameters used were adopted from [3] as shown in Table 3. The simulation topology in [3] was adopted as shown in Fig. 7, which consists of one Base Station (BS), 35 Subscriber Stations (SS) distributed around the BS, and an application server. The traffics are generated from the server, which provides four traffics each from a different application. We assume that each user traffic is carried by one SS and that each user can only use one type of traffic in a given time. The traffics are prioritized according to their QoS requirements in the following order:

UGS → rtPS → nrtPS → BE.

Figure 8 shows the average delay on SSs for LAWRR and PLAS for the rtPS class. The figure demonstrates that the PLAS achieves better result compared to the other scheme. This is as a result of the increase in the service rate that allow majority of packets to be sent and hence lead to a smaller number of packets being delayed in each around.

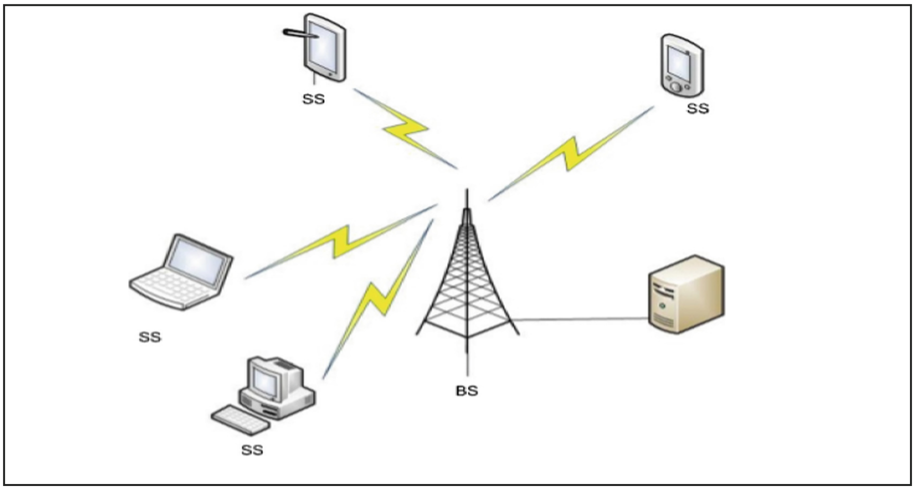

Fig. 7. Shows simulation network topology

Table 3. Simulation parameters

Parameter	Value
Base station Freq.	2.5 GHz
Duplexing mode	TDD
Bandwidth	5 Mbps
Frame length	5 ms
Cyclic prefix duration	11.43 μs
Basic symbol	91.43 μs
FFT	1,024
PHY	OFDM
DL permutation zone	PUSC
MAC PDU length	Variable
Fragmentation	Enable
ARQ and packing	Disable
DL-UL MAPS	Variable

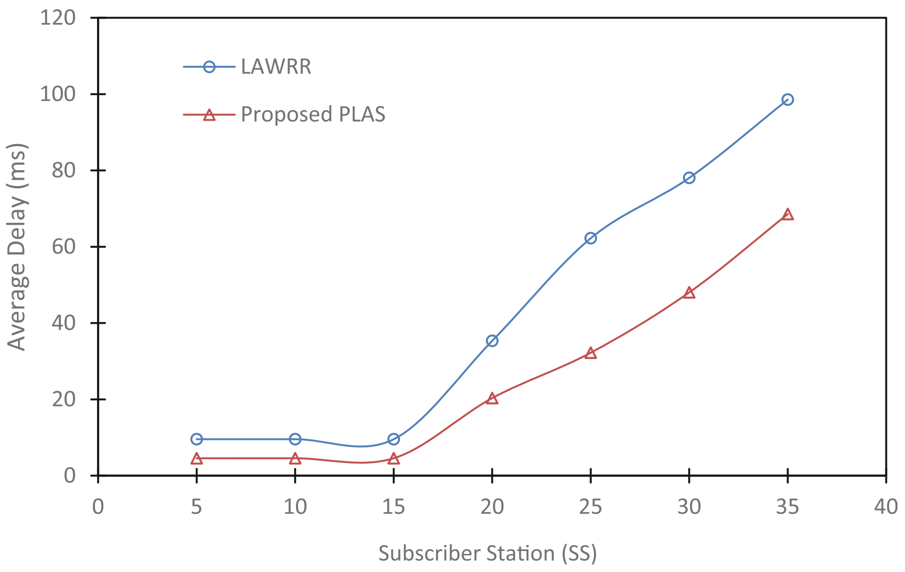

Fig. 8. Shows average delay for the rtPS class on SS

5 Conclusions

In this paper, a PLAS is proposed to increase the service rate of real time traffics in WiMAX networks. The algorithm introduces a mechanism that prioritizes real time over non-real time traffics. Simulation experiments were carried out to evaluate the performance of the proposed PLAS and the LAWRR algorithms. The results demonstrated that PLAS yields better performance than LAWRR in terms of average delay.

Acknowledgement. The authors would like to appreciate the Tertiary Education Trust Fund (TETFund) Nigeria for supporting this research work through the National Research Fund (NRF) Grant No: CC – STI.

References

1. Chakchai, S., Raj, J., Adel-Karim, T.: Scheduling in IEEE 802.16e mobile WiMAX networks: key issues and a survey. IEEE J. Sel. Areas Commun. **27**, 156–171 (2009)
2. Audace, M., Saadi. B., Lami, C.F.: A priority-weighted round robin scheduling strategy for a WBAN based healthcare monitoring system. In: 13th IEEE Consumer Communications & Networking Conference (CCNC), pp. 224–229 (2016)
3. Ibrahim, S., Shamala, S., Azmin, J., Zuriati, Z.: A load-aware weighted round-robin algorithm for IEEE 802.16 networks. EURASIP J. Wirel. Commun. Netw. **2014**, 1–12 (2014)

4. Mohamed-el-Amine, B., Abdelhafid, A., Lorenz, P.: Adaptive scheduling mechanism for IPTV over WiMAX IEEE 802.16j networks. Int. J. Commun. Syst. (2012) **27**, 1009–1019 (2014)
5. Zuber, P., Uperia, D.: Design and implementation of low latency weighted round robin (LLWRR) scheduling for high speed networks. Int. J. Wirel. Mob. Netw. (IJWMN) **6**, 59–71 (2014)
6. Claudio, C., Luciano, L., Enzo, M.: Quality of service support in IEEE 802.16 networks. IEEE Netw. Mag **20**, 50–55 (2006)
7. Alexander, S., Olli, A., Timo, H.: Scheduling solution for IEEE 802.16 base station. Int. J. Comput. Telecommun. Netw. **52**, 96–115 (2008)
8. Chih-Peng, L., Jenhui, C., Hsing-Lung, C.: An efficient bandwidth allocation algorithm for real-time VBR stream transmission under IEEE 802.16 wireless networks. J. Netw. Comput. Appl. **33**, 467–476 (2010)
9. Mardini, W., Abu Alfoul, M.M.: Modified WRR scheduling algorithm for WiMAX networks. Netw. Protoc. Algorithms J. **3**, 24–53 (2011)
10. Saha, D., Mukherjee, S., Tripathi, S.: Carry-over round robin: a simple cell scheduling mechanism for ATM networks. IEEE/ACM Trans. Netw. **6**, 779–796 (1996)
11. Manoli, K., Stefanos, S., Costas, C.: IEEE J. Sel. Areas Commun. **9**, 1265–1279 (1991)