EURASIP Journal on
Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

A load-aware weighted round-robin algorithm for IEEE 802.16 networks

Ibrahim Saidu[1], Shamala Subramaniam[2*], Azmi Jaafar[1] and Zuriati Ahmad Zukarnain[1]

Abstract

The IEEE 802.16 standard was designed to provide quality-of-service (QoS) guarantees for various classes of traffic with diverse QoS requirements. Packet-scheduling algorithms play a critical role in providing such guarantees. Weighted round robin (WRR) is one of the most commonly used scheduling algorithms, because of its simplicity and low computational overhead. However, it suffers from performance degradation under bursty traffic conditions because of the static weights used to determine packet transmissions. We propose a new packet-scheduling discipline for downlink traffic in 802.16 networks to improve performance in such situations. It dynamically determines the weight of each queue in the various classes based on the current traffic characteristics using the static WRR weight at the beginning of each base station round. The goal is not only to reduce the average delay and packet loss but also to improve average throughput. Simulations are used to evaluate the performance of the proposed algorithm load-aware weighted round robin (LAWRR), and show that it reduces average delay and packet loss, as well as it improves average throughput compared with WRR. The effectiveness of LAWRR running under fixed buffer sizes is also investigated.

Keywords: Load-aware WRR; Dynamic weighting; Bursty traffic; IEEE 802.16

1 Introduction

The IEEE 802.16 standard [1], popularly known as Worldwide Interoperability for Microwave Access (WiMAX), addresses broadband access technology for wireless metropolitan area networks (WMANs). The standard sets out two specifications, the physical (PHY) and media access control (MAC) layers. It has several advantages, including ease and cost of deployment, first-mile/last-mile access, and quality-of-service (QoS) support for multimedia applications at the MAC layer [2]. Because multimedia applications must support different types of traffic simultaneously, each of which has different QoS requirements from the network, such as bandwidth, delay, jitter, and packet loss, providing QoS to these traffic classes represents a challenge. Therefore, scheduling algorithms are essential to providing the required level of network QoS [3-5].

Scheduling algorithms provide a mechanism for allocating often scarce wireless resources among subscriber stations (SSs) and determining their transmission order. Several such algorithms have been proposed to support QoS in wired and wireless networks [6]. Some are classical algorithms that were not specifically designed for IEEE 802.16 but have been applied practically [7-10]. Among these, weighted round robin (WRR), which differentiates services among various traffic classes, is the most commonly used algorithm, because of its simplicity and low computational cost. WRR uses static weights to differentiate QoS requirements for the various service classes. It starts by classifying packets according to service class and followed by assigning them into queues. Each queue within a class is assigned a fixed weight, which specifies the number of packets to be transmitted in a single round. Because the algorithm sends fixed numbers of packets, bursts of input traffic lead to increasing queue sizes. Moreover, packet loss may occur in the presence of heavy input bursts and hence reduce throughput. Therefore, the use of fixed weights is adequate for constant-rate classes because of its fixed weighting priority for each queue. Although WRR distinguishes classes according to their QoS requirements [11], it performs poorly under bursty traffic [12,13].

In this paper, we propose a new packet-scheduling algorithm, *load-aware weighted round robin* (LAWRR), for

*Correspondence: shamala_ks@upm.edu.my
[2]Sports Academy, Universiti Putra Malaysia, 43400 UPM Serdang, Selangor DE, Malaysia
Full list of author information is available at the end of the article

802.16 networks in the downlink direction to improve upon the efficiency of WRR. Our algorithm adaptively employs dynamic weighting to adjust the weight of each queue according to the traffic characteristics and the WRR static weight. The goal is to reduce the delay and packet loss due to input burst traffic as well as to improve throughput. The algorithm is evaluated through discrete-event simulations. The performance of the proposed algorithm is compared and evaluated by means of simulations against WRR [14]. The results show that LAWRR outperforms WRR in terms of delay and packet loss as well as throughput. The effectiveness of our approach is also examined when running with finite buffer sizes, in which it again proves to be successful.

This paper is organized as follows: Section 2 briefly reviews the IEEE 802.16 standard, and Section 3 addresses related work on scheduling algorithms. In Section 4, we describe our proposed algorithm. Section 5 presents the performance evaluations, and we summarize in Section 6.

2 IEEE 802.16

The standard [1] defines two layers to support fixed WMAN. The MAC layer sits atop the PHY layer and mediates between it and the layers above. The protocol that operates the MAC layer performs the main tasks of the standard, such as QoS provisioning, connection admission control (CAC), bandwidth allocation, and scheduling. It supports two modes of operation: point-to-multipoint (PMP) and mesh. The former is a cellular-like structure that supports communication between a base station (BS) and a set of SSs in broadcast fashion. The BS is the central controller, regulating all communications between itself and a set of SSs. The two paths of communication between the BS and the SSs are the uplink ('UP'; from SS to BS) and downlink ('DL'; from BS to SS) directions. In contrast, mesh mode supports multihop communication between SSs. In this paper, we consider PMP as the main operational mode.

The PHY layer is responsible for transmitting bits over the wireless channel by means of the adaptive modulation and coding (AMC) technique. AMC supports two transmission modes, frequency-division duplexing (FDD) and time-division duplexing (TDD). In FDD mode, uplink and downlink data are sent on different frequencies. In contrast, in TDD mode, both UP and DL data are sent using the same frequency but in different time slices. Both duplexing modes operate in a frame format. Each frame is partitioned into DL and UP subframes. The DL subframe is used by the BS to transmit data and manage messages to an SS, while the UP subframe is used by all SSs to transmit data.

The MAC layer is a connection-oriented protocol that has the advantage of controlling network resource sharing among individual connections. The protocol maps both connected and connectionless traffic to a unique connection identifier (CID). If traffic coming from an upper layer arrives at the MAC layer, the SS attempts to establish a connection with the BS. The BS employs a call admission control (CAC) scheme that checks whether the resources available can ensure QoS for the new connection while maintaining the QoS guarantees for the existing users. With the acceptance of a new connection, the BS responds to the SS with a CID to use for the UP and DL traffic. Once a connection is set, the SS can request bandwidth from the BS. The BS grants bandwidth using the grant per subscriber station (GPSS) approach. Once the SS receives bandwidth from the BS, its packet scheduler distributes the bandwidth among its own active connections. The CAC and request-grant bandwidth allocation components of the BS provide support to different applications with various QoS requirements. The 802.16 standard partitions applications into service classes as follows.

The *unsolicited grant service* (UGS) periodically generates constant-size data packets for real-time traffic such as voice over Internet Protocol (VoIP) without silence suppression. UGS is sensitive to transmission delays, and the BS allocates grants to the SSs in an unsolicited fashion using the maximum sustained traffic rate (MSTR), traffic priority, and maximum latency tolerance as QoS requirements.

The *real-time polling service* (rtPS) generates variable-size data packets for real-time traffic such as MPEG video. It has less stringent delay requirements and is periodically polled by the BS for each SS to individually determine its bandwidth requirement. Its mandatory QoS specifications are the minimum reserved traffic rate (MRTR), MSTR, traffic priority, and maximum latency tolerance.

The *extended real-time polling service* (ertPS) generates variable-size data packets for real-time traffic such as VoIP with silence suppression. It combines features of both UGS and rtPS and has strict, guaranteed delay requirements and provides unicast grants in an unsolicited manner by the BS, as with UGS. Because UGS grants are of constant size whereas ertPS grants vary in size, an SS can request a change of its bandwidth grant to suit its requirements. The ertPS QoS requirements are MRTR, MSTR, traffic priority, maximum latency tolerance, and delay jitter tolerance.

The *non-real-time polling service* (nrtPS) generates variable-size data packets for non-real-time traffic such as FTP. It has minimum bandwidth requirements that are delay tolerant. It is polled by the BS in order for each SS to state its desired bandwidth. The QoS requirements are MRTR, MSTR, and traffic priority.

The *best-effort* (BE) *service* is designed to support traffic for which delay and throughput are not guaranteed, such as HTTP. It requests bandwidth through contention request opportunities and unicast request opportunities.

Challenges constantly emerge as a result of the evolving demands of applications in IEEE 802.16 networks. These include security issues, path loss, handover, and scheduling. The latter is a critical component in providing QoS to various applications with diverse QoS requirements. The QoS scheduling services and their parameters are defined in the IEEE 802.16 MAC layer, but the actual algorithms are left unspecified [15]. Hence, the scheduling algorithms remain vendor specific. There are three scheduling algorithms that must be implemented in an IEEE 802.16 network: the BS accounts for two, including UP and DL scheduling, while SSs have only an UL scheduling algorithm. Here, we consider the DL algorithm.

3 Existing algorithms

Several scheduling algorithms have been proposed to address the problems that affect applications running under different IEEE 802.16 service classes. This section provides an overview of some major scheduling algorithms that have been applied in and proposed for 802.16 networks.

First-in, first-out (FIFO) scheduling is considered the earliest and simplest algorithm, as it accepts packets from all input traffic classes. These are added to a single queue upon arrival and then serviced to the output links on a first-come, first-served basis. It is thus suitable for UGS traffic. However, a FIFO scheduler cannot service one class of traffic differently from any other [16].

The round-robin (RR) algorithm [17] is another simple approach designed for time-sharing systems [18], which considers priority on queues. RR starts by classifying packets into low- and high-priority service classes. Once mapped, these packets are assigned to dedicated queues. Each nonempty queue is then allowed to send a packet, starting with the highest priority class, without any further priority assignment by the scheduler. RR serves all nonempty queues in cyclical order; once the queue is served from a given class in the current service round, it is not served again until all unserved queues have been served within a single round. If there are still nonempty queues, the algorithm moves to a next round until they are emptied. RR is therefore fair when equal packet lengths are used but unfair when variable packet lengths are used by different queues. Moreover, it cannot guarantee different QoS requirements for different service classes [19].

WRR is a modification of RR proposed by [14] that considers the QoS requirements for each class. Static weights are used to distinguish the QoS requirements of each class and also to determine its required bandwidth. Each class has a list of queues, and each of these has an associated weight counter. If a queue is picked for service, its (static) weight is assigned to its counter, which specifies the number of packets to be transmitted in a round. The weight of each queue is calculated by considering the QoS requirements of each class. The various QoS parameters used are traffic priority, MSTR, and MRTR [20,21]. The algorithm serves each queue in RR fashion from highest to lowest priority class. Packets from each queue are served in each round from those whose weights are nonnegative. If a packet is sent, a queue weight counter is decremented. Packets continue to be sent until the counter reaches zero or the queues in all classes are empty. Finally, all queue counters are reset to their weight values. Therefore, WRR is fair when equal weights and equal packets are used by all the queues in each class. However, the use of static weighting to serve classes with variable priority levels under input burst traffic may lead to rapid growth of their queues and, consequently, to further packet delay as well as packet loss. It is also unfair to queues with variable packet lengths. Furthermore, unlike real-time queues, which are given higher weights, non-real-time queues are given lower weights [20,22,23], thus imposing a delay on lower priority classes.

Modified weighted round robin (MWRR) is a variation proposed by [11] to overcome this penalty for lower priority classes. It starts by computing the WRR weight for each queue based on priority and the number of all nonempty queues. This weighting permits each queue to transmit a certain number of packets in a single service round. Thus, the total number of packets a WRR scheduler can deliver per service round is fixed. The MWRR scheduler multiplies each WRR weight counter by a constant integer value in order to increase the service round. This algorithm reduces the average delay and increases the average throughput, particularly for lower priority classes, by lengthening the size of the service round over WRR. However, the multiplier used is static, which may lead to either increase in delay or decrease in throughput when it is inappropriately chosen.

Adaptive weighted round robin (AWRR) is yet another variant, proposed by [24], again to avoid the problem of starvation of lower priority classes. It uses two schedulers: an input scheduler aims to prioritize video streams with superior quality (HD and SD) and consists of a high-priority (HP) buffer for, e.g., UGS, ertPS, and rtPS, and a low-priority (LB) buffer, which includes rtPS-web-TV, rtPS-mobile-TV, nrtPS, and BE. On the other hand, the objective of the output scheduler is to regulate data flows and manage all the service classes. Both schedulers use AWRR to adjust the weighting of the service

classes. A threshold value is set for each class, which triggers dynamic weight adjustment whenever a threshold is exceeded. The HP queues have lower thresholds than do the LP queues. The input scheduler controls the weights of the HP and LP queues based on the HP traffic load and buffer size. Control of the weights from the queue allows the algorithm to achieve minimum throughput of BE traffic under conditions of network stress and also to give preferential treatment to HP traffic. The dynamic weight is calculated using an algorithm in which the weights need to be positive to ensure minimum throughput for LP traffic, and an arbitrary constant is used to favor HP traffic classes such as UGS and rtPS. However, it employs a complex calculation to compute the weights and applied in WiMAX multihop networks.

To date, none of these algorithms has proved capable of further reducing delays and packet loss while improving the throughput in IEEE 802.16 networks by considering traffic load in each queue. The packet-scheduling algorithm that we propose differs from these efforts in that it dynamically adjusts the static weights according to the traffic load of each queue in each service class.

4 Proposed algorithm

In this section, a new modification of the WRR scheduling algorithm, which we call LAWRR, is described. First, however, the shortcomings of the WRR discipline are presented. WRR is a modification of the RR algorithm that services queues according to static weightings. The weights are determined according to the priority of the service classes as UGS > rtPS > nrtPS > BE. The weight of each class is computed based on its minimum bandwidth requirements. The UGS and BE classes have MSTR as their bandwidth requirement. UGS can use MSTR in the computation of its weight because it is allowed by the standard to transmit at a constant rate, but BE is not allowed to transmit unless the higher priority classes have nothing to send. Finally, rtPS and nrtPS have MSTR and MRTR as their bandwidth requirements, since the weight computation is based on MRTR (see [20]).

Therefore, static weighting is suitable for UGS traffic because of its fixed-weight priority level. However, it is not suitable for the rtPS and nrtPS classes, which have variably weighted priorities. It is also not suitable for BE, which has only MSTR as its bandwidth parameter and is not allowed to transmit at a constant rate. When there are queues in each class, a weight counter is assigned to each. If a queue is picked for service, its static weight is calculated using Equation 1 below and assigned to its weight counter, which specifies the number of packets an SS is allowed to send. If a packet is sent, then the counter is decremented by 1. It moves to the next service round after all queues are served in the current round. It then continues to serve each queue in each class in RR fashion until the counter reaches zero or the queues are exhausted from all classes. Finally, the counters are assigned to their weight values again:

$$W_i = \frac{\mathrm{MRTR}_i}{\sum_{i=1}^{N} \mathrm{MRTR}_i}, \tag{1}$$

where W_i is the WRR static weight of queue i, MRTR_i is the minimum reserved traffic rate of queue i, which reflects its QoS requirements, and N is the total number of queues in a class.

Based on this scenario, burstiness in the input traffic presents a great challenge of creating delay and packet loss as well as lowering the throughput in IEEE 802.16 networks, which leads to performance degradation. As can be seen, a class may have more packets arriving than its associated weight counter value, cannot be handled in a single round, as shown in Figure 1. This figure shows that the WRR scheduler assigned a static weight of 2 to each queue weight counter, which means that only two packets can be transmitted from each queue. Since the counter decreases by 1 for each packet, it reaches zero before all packets in the queue have been transmitted, as shown in Figure 2, and hence, the WRR scheduler for this queue is suspended until the next counter reset

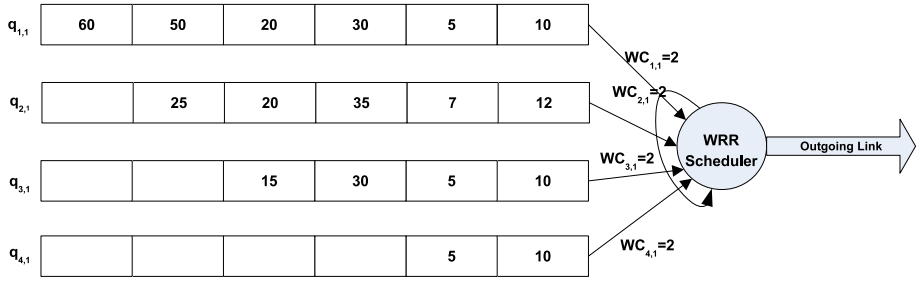

Figure 1 Illustration of the WRR algorithm. Queue and scheduler states are shown.

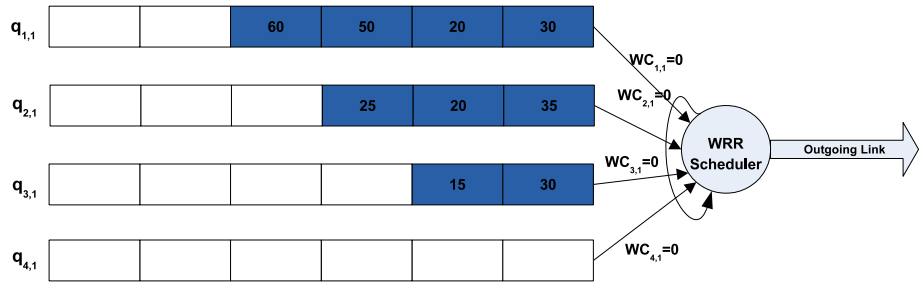

Figure 2 State of the WRR algorithm before a counter reset. The queues are shown after the weights have been exhausted.

[14]. Figure 2 shows that before the next reset, a total of nine packets remain in the queues waiting for transmission. Therefore, packets waiting in the queues are further delayed. Moreover, with heavy input burst traffic, the queue length will continue to grow rapidly, which also tends to cause an increase in packet loss. Ultimately, there will be a reduction of throughput as a result of the packet loss.

For example, if the $q_{i,1}$ represents queue i ($i = 1, 2, \ldots, n$) at round 1, then the scheduler in Figure 1 schedules packets as:

$$q_{1,1} \rightarrow q_{2,1} \rightarrow q_{3,1} \rightarrow q_{4,1} \rightarrow q_{1,1} \rightarrow q_{2,1} \rightarrow q_{3,1} \rightarrow q_{4,1}.$$

Completion of the above sequence constitutes a round. The scheduler thus transmits two packets from each queue in a single round.

To address the shortcomings of WRR described above, we propose an algorithm that changes the static weight of each class into a dynamic weight. These are computed dynamically according to the traffic characteristics of the queue and the static weight of each class at the beginning of every counter reset. These keep track of the traffic variations of each queue to reduce the delay and packet loss, as well as to improve the throughput in an IEEE 802.16 network under input burst traffic.

The dynamic weight computation is a four-step process. First, we compute the load variance: assume that at round k, the load of queue i in an IEEE 802.16 network is given by $\text{load}_{i,k}$, where $1 \leq i \leq n$ for a class with n queues. The variance of the queues at round k (α_k) is calculated as:

$$\alpha_k = \frac{1}{n} \sum_{i=1}^{n} \left(\text{load}_{i,k} - \langle \text{load}_k \rangle\right)^2 , \tag{2}$$

where $\langle \text{load}_k \rangle$ is the mean from round k,

$$\langle \text{load}_k \rangle = \frac{1}{n} \sum_{i=1}^{n} \text{load}_{i,k} . \tag{3}$$

Second, the root mean square (rms) errors at round k (β_k) are determined as:

$$\beta_k = \sqrt{\alpha_k} . \tag{4}$$

Third, the dynamic coefficient of queue i at round k ($\omega_{i,k}$) is computed as:

$$\omega_{i,k} = \left\lceil \frac{\text{load}_{i,k}}{\beta_k + 1} \right\rceil . \tag{5}$$

We take the ceiling ($\lceil y \rceil$ the minimum integer $\leq y$) because $\omega_{i,k}$ is the dynamic coefficient of the dynamic weight and must be an integer, because the dynamic weight specifies the number of packets the scheduler is allowed to transmit from the queue in one counter reset.

Finally, the dynamic weight of queue i in round k ($w_{i,k}$) is given by:

$$w_{i,k} = \omega_{i,k} W_i . \tag{6}$$

It can be seen from this expression that very large values of $w_{i,k}$ will extend the counter reset time. To avoid this, $w_{i,k}$ needs to be made a small integer by dividing by a constant common to all the queues and rounding off the result to an integer value. This will yield an appropriate weight for each queue. For example, assume that there are three queues in the first round and that their weights $w_{1,1}$, $w_{1,2}$, and $w_{1,3}$ are computed as 201, 203, and 243, respectively. After dividing these values by 40, we obtain 5, 6, and 7 as more appropriate values of $w_{1,1}$, $w_{1,2}$, and $w_{1,3}$, respectively. We adopted the idea of using an appropriate constant value in [25]. The author used it in order to avoid a large quantization error from large divisor.

The following example demonstrates the method used to compute the dynamic weights based on Figure 1 and Algorithm 1. First, we consider the proposed algorithm based on Figure 1 and assume the static weighting. Table 1 shows the computation of the dynamic weight, including the variance ($\alpha_1 = 3{,}450.1875$), rms error rate ($\beta_1 = 58.7383$), dynamic weight coefficients (column 3),

and dynamic weight of each queue (column 5) while assuming $k = 1$. From the table, our dynamic weights are obtained and represented as $w_{1,1} : w_{2,1} : w_{3,1} : w_{4,1} = 6 : 4 : 2 : 2$. Next, we assign these weights to respective weight counters, $\mathrm{WC}_{1,1} = 6$, $\mathrm{WC}_{2,1} = 4$, $\mathrm{WC}_{3,1} = 2$, and $\mathrm{WC}_{4,1} = 2$, which indicate the number of packets to transmit in each queue, as shown in Figure 2.

Algorithm 1: LAWRR scheduling

```
n ← number of connection queues
q_{i,k} ← current connection of queue i at round k
w_{i,k} ← dynamic weight counter of queue i at round k
WC_{i,k} ← weight counter of queue i round k
k ← current RR round
WC_{i,k} ← 0 (i = 0, 1, 2, ..., n − 1)
k ← 0
for i ← 0 to n − 1 do
    if q_{i,k} ≠ NULL then
        compute w_{i,k} using Equation 6
        WC_{i,k} ← w_{i,k}
        if q_{i,k} ≠ NULL and WC_{i,k} ≠ NULL then
            transmit packets from q_{i,k} using WRR
        else
            k ← k + 1
end
```

Compared with Figure 1, Figure 3 shows that the LAWRR scheduler has dynamically adjusted the static weight for $q_{1,1}$ and $q_{2,1}$ according to their traffic load characteristics. After assigning a dynamic weight to the weight counter, LAWRR begins serving each queue starting from the head of the first queue to the last queue. It continues to serve each in cyclical fashion until the queue is empty or the weight counter reaches zero. The LAWRR scheduler for this queue is then suspended until the next counter reset. Figure 4 shows that, before the next counter reset, a total of three packets are still queued. In this figure, as compared with the WRR scheduler in Figure 2, the LAWRR scheduler reduced the total number of delayed packets from nine to three.

Table 1 Computation of dynamic weights

	$\mathbf{load}_{i,1}$	$(\mathbf{load}_{i,1} - \langle \mathbf{load}_1 \rangle)^2$	$\omega_{i,1}$	$w_{i,1}$
$q_{i,1}$				
$q_{1,1}$	175	7,700.0625	3	6
$q_{2,1}$	99	138.0625	2	4
$q_{3,1}$	60	742.5625	1	2
$q_{4,1}$	15	5,220.0625	1	2
Total	349	13,800.75		
Statistics	$\langle \mathrm{load}_1 \rangle = 87.25$	$\alpha_1 = 3{,}450.1875$, $\beta_1 = 58.7383$		

By adaptively adjusting weights in proportion to the traffic load characteristics and static weight of each queue, the LAWRR scheduler is flexible in its ability to transmit packets proportionally to the traffic arrival rate. In other words, when packets are arriving in a queue belonging to a particular connection at a high rate, the dynamic weight assigned to the queue will allow it to send more packets than the static weighting, which may lead to a reduction of delay and packet loss of the rtPS class, as well as improving the throughput of the rtPS, nrtPS and BE classes.

Our proposed scheduler (Figure 3) schedules packets as:

$$\begin{aligned} q_{1,1} \to q_{2,1} \to q_{3,1} \to q_{4,1} \to q_{1,1} \to q_{2,1} \to q_{3,1} \\ \to q_{4,1} \to q_{1,1} \to q_{2,1} \to q_{1,1} \to q_{2,1} \\ \to q_{1,1} \to q_{1,1} \,. \end{aligned}$$

It thus transmits six packets for $q_{1,1}$, four for $q_{2,1}$, and two each for $q_{3,1}$ and $q_{4,1}$.

Because the dynamic weight alone cannot be used to determine the bandwidth requirements of each service class, the allocated bandwidth for each class is calculated as follows.

First, let $w_{i,k}$ be the dynamic weight associated with the ith queue in round k and $M_{i,k}$ be the average packet length of the ith queue in round k. The average number of bits transmitted during each round is given by:

$$N_{b,k} = w_{i,k} M_{i,k} \,. \tag{7}$$

Given m queues, the average number of bits sent during round k is:

$$N_{b,k}^{m} = \sum_{i=0}^{m} w_{i,k} M_{i,k} \,. \tag{8}$$

Next, we consider the average packet length and weights of the m queues. The outgoing link bandwidth required for the jth queue during round k is obtained as:

$$B_{j,k} = \frac{w_{j,k} M_{j,k}}{\sum_{i=0}^{m} w_{i,k} M_{i,k}} \,. \tag{9}$$

Finally, we assume that a service class contains N_j nonempty queues and that each class is associated with a queue of the LWRR scheduler. The bandwidth allocated for queue j of service class x during round k is

$$B_{j,k}^{x} = \frac{w_{j,k} M_{j,k}}{N_j \sum_{i=0}^{m} w_{i,k} M_{i,k}} \,. \tag{10}$$

This is the minimum amount of bandwidth needed to satisfy the QoS requirements of all queues in service class x for round k.

5 Performance

In this section, we compare the performance of WRR [14] with the proposed LAWRR scheduling algorithm. We

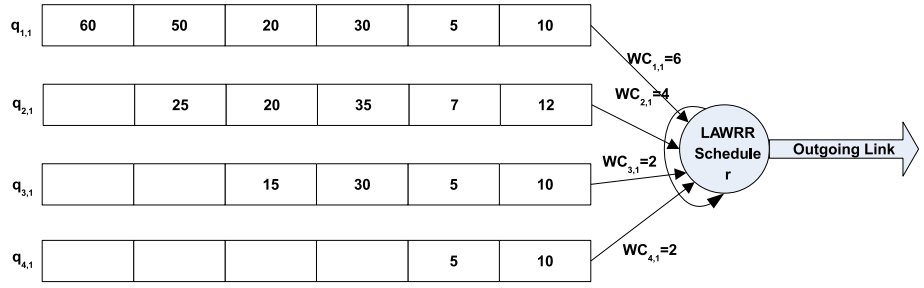

Figure 3 Illustration of the LAWRR algorithm. Compare with Figure 1.

have developed a discrete-event simulation code and used it to conduct simulations to evaluate the performance of our proposed algorithm.The simulation parameters are presented in Table 2.

5.1 Traffic model and simulation parameters

5.1.1 Traffic models

Four traffic models are used in the simulations, adopted from [1].

VoIP (class 2) VoIP as a traffic source was modeled with an exponential distribution of ON and OFF periods (or talk spurts and silence) with the source alternating between ON and OFF periods. Several encoding schemes are available for voice, such as G.723.1, G.728, G.729, and adaptive multi-rate (AMR), which have different bandwidth requirements. We modeled the ON and OFF periods with the AMR codec (Table 3).

Video conferencing (class 2) This source has both audio and video components, each with its own bandwidth requirements. For example, the audio component requires between 16 and 64 kbps, whereas the video component requires from 320 kbps to 1 Mbps. We implement a typical business-quality videoconference that runs at 384 kbps and can deliver TV-quality video at 25 to 30 frames per second [10] (see Table 4).

***FTP* (*class 5*)** This traffic was modeled as an exponential source with a mean of 0.0277 s and constant packet size of 150 B. The source generates a geometric number of packets with a mean of 25 packets and exponential reading time with a mean 5 s per session. According to the packet size distribution, 76% of the files are transferred using a maximum transmission unit (MTU) of 1500 B and 24% of the files are transferred using an MTU of 576 B. These packets also have a 40 B IP header overhead.

***HTTP* (*class 5*)** This traffic was modeled as an exponential source with mean 0.0277 s, where a source generates constant packet lengths of 150 B. In each HTTP session, there are a number of packet calls, and within each call, there are a number of packets. The number of calls within a session and the number of packets within a call follow a geometric distribution with means of 5 and 25 packets, respectively. The calls are separated by a geometric reading time with a mean of 5 s.

5.1.2 Simulation scenario

The simulated network consists of a BS and 35 uniformly distributed SSs, as shown in Figure 5. The traffic source is an application server that provides four types of applications, each of which is associated with one traffic type. We assume that each SS carries the traffic of a single user and that each user can only use one type of traffic in a given time. The traffic is prioritized according to service

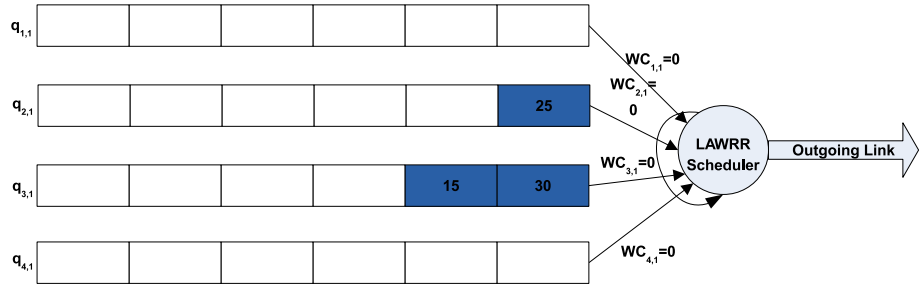

Figure 4 State of the LAWRR algorithm before the next counter reset.

Table 2 Simulation parameters in [10]

Parameter	Value
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	1,024
PHY	OFDMA
DL permutation zone	PUSC
MAC PDU length	Variable
Fragmentation	Enable
ARQ and packing	Disable
DL-UL MAPS	Variable

class, with UGS given the highest priority and BE given the lowest.

5.1.3 Performance metrics

Four measures were used to characterize the performance of the three scheduling algorithms over the WiMAX network link when different applications are supported.

Average throughput The number of packets received by an SS per unit time in kilobits per second:

$$T_{\text{avg}} = \frac{1}{S}\sum_{i=0}^{n} P_{\text{rec}}(i)\,, \tag{11}$$

where $P_{\text{rec}}(i)$ is the packet i received at the SS and S is the total simulation time of the experiment.

Average queuing delay The time in milliseconds between the arrival and departure of a packet from the queue:

$$D_{\text{avg}} = \frac{1}{n}\sum_{i=0}^{n}\left[t_{\text{dep}}(i) - t_{\text{arr}}(i)\right]\,, \tag{12}$$

where $t_{\text{arr}}(i)$ is the arrival time of packet i in the queue, $t_{\text{dep}}(i)$ is the departure time of packet i from the queue,

Table 3 VoIP traffic parameters [26,27]

Parameter	Distribution	Value
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 [10]

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

and n is the total number of packets that arrived in the queue.

Packet loss The percentage of packets per SS dropped from the queue out of all that arrived:

$$P_{\text{loss}} = \frac{\sum_{i=0}^{m} P_{\text{drop}}(i)}{\sum_{i=0}^{n} P_{\text{arr}}(i)}\,, \tag{13}$$

where $P_{\text{drop}}(i)$ is the number of packets i dropped from the queue and $P_{\text{arr}}(i)$ is the number that arrived.

Packet drop ratio The ratio of the cumulative number of dropped packets (CDP) to cumulative generated packets (CGP), used to examine the effectiveness of LAWRR running with finite buffers:

$$\text{PDR} = \text{CDP}/\text{CGP}\,. \tag{14}$$

5.1.4 Results and discussion

The simulations were carried out under various network conditions such as the low or heavy traffic load and number of SSs in the network.

Low traffic Figures 6 and 7 show the packet delay and loss for different numbers of connections for the WRR and LAWRR scheduling algorithms for the rtPS class. LAWRR performs similarly to WRR for 0 to 25 SSs because of the low traffic load. When the load increases to 25 to 35 SSs, LAWRR performs better because of the proportionate increase in service rounds with traffic load characteristics. It is found that there are 27.65% and 25.77% improvements from LAWRR compared with WRR.

Figure 8 shows the average WRR and LAWRR throughput for the rtPS class. Again, for 0 to 25 SSs, the two algorithms perform similarly because of the low traffic. But from 25 to 35 SSs, LAWRR outperforms WRR because our proposed algorithm reduces packet loss due to bursty traffic while increasing the number transmitted. LAWRR improves the throughput by 41.75% over WRR.

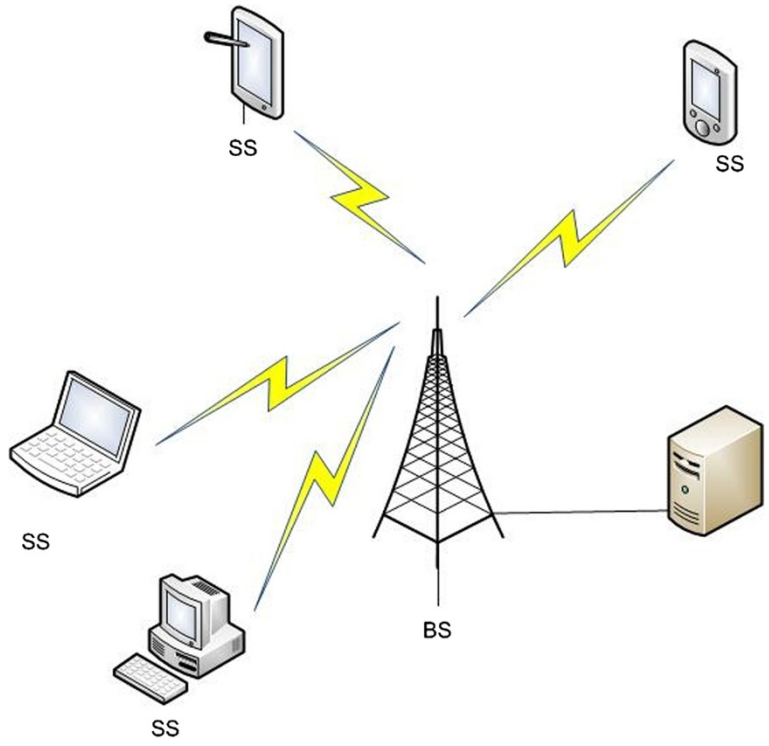

Figure 5 Simulation network topology.

Figure 9 shows the same for the nrtPS class. LAWRR outperforms WRR throughout. This is due to the proportionate increase in the number of packets to be transmitted in a round according to the current load characteristics. LAWRR improves the throughput by 57.75% compared with WRR.

Figure 10 similarly shows the average throughput for the BE class. LAWRR is better than WRR for 0 to 20 SSs but has similar performance from 20 to 35 SSs. This is a result of the opportunity given to BE when traffic is light and its service round is in accordance with the traffic load characteristics of each queue, resulting in high packet transmission. LAWRR improves the throughput by 31% compared with WRR.

High traffic Figure 11 shows the packet delay for different numbers of connections for WRR and LAWRR for the rtPS class. This figure shows that LAWRR outperforms

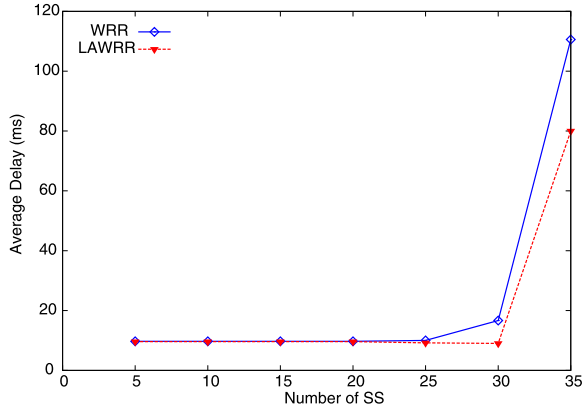

Figure 6 Average delay per SS for the rtPS class. It shows the packet delay and loss for different numbers of connections for the WRR and LAWRR scheduling algorithms for the rtPS class.

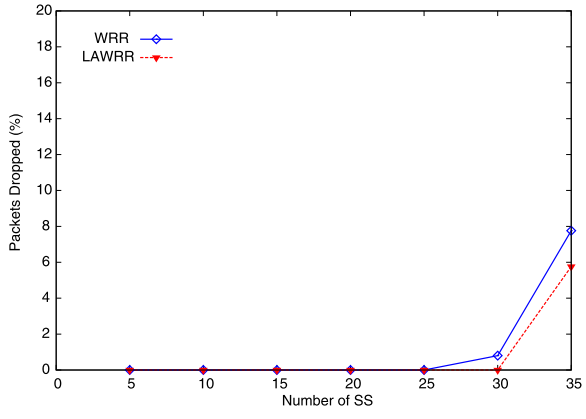

Figure 7 Packet loss per SS for the rtPS class. It shows the packet delay and loss for different numbers of connections for the WRR and LAWRR scheduling algorithms for the rtPS class.

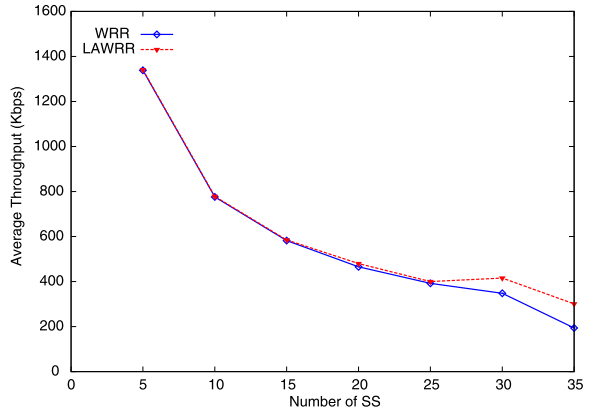

Figure 8 Average throughput per SS for the rtPS class. For 0 to 25 SSs, the two algorithms perform similarly because of the low traffic. But from 25 to 35 SSs, LAWRR outperforms WRR because our proposed algorithm reduces packet loss due to bursty traffic while increasing the number transmitted.

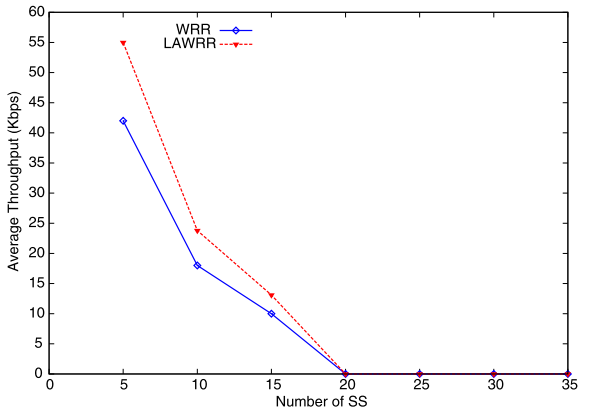

Figure 10 Average throughput per SS for the BE class. LAWRR is better than WRR for 0 to 20 SSs but has similar performance from 20 to 35 SSs. This is a result of the opportunity given to BE when traffic is light and its service round is in accordance with the traffic load characteristics of each queue, resulting in high packet transmission.

WRR. The reason is a proportionate increase in service rounds with traffic load, which in turn allows more packets to be transmitted and, thus, leads to fewer packets being delayed in each round. LAWRR here demonstrates a 29.59% improvement over WRR.

Figure 12 similarly shows the packet loss. LAWRR and WRR are competitive for 0 to 15 SSs, because the service rounds for each algorithm are the same and the traffic is light, but LAWRR performs better for 15 to 35 SSs because the service rounds increase under high traffic loads, reducing the number of previously delayed packets. LAWRR improves on the performance of WRR by 27.15%.

Figure 13 shows the average WRR and LAWRR throughput for the rtPS class. For 0 to 15 SSs, the algorithms are similar because there is no packets loss, given the low traffic load. But from 15 to 35 SSs, LAWRR outperforms WRR because it increases the number of packets transmitted in response to bursty traffic. LAWRR improves the throughput by 41.75% compared with WRR.

Figure 14 shows the same for the nrtPS class. LAWRR performs better than WRR throughout. This is because the number of packets to be transmitted in a round is increased in proportion to the current load characteristics, with a reduction of packet loss. LAWRR improves the throughput by 57.75% compared with WRR.

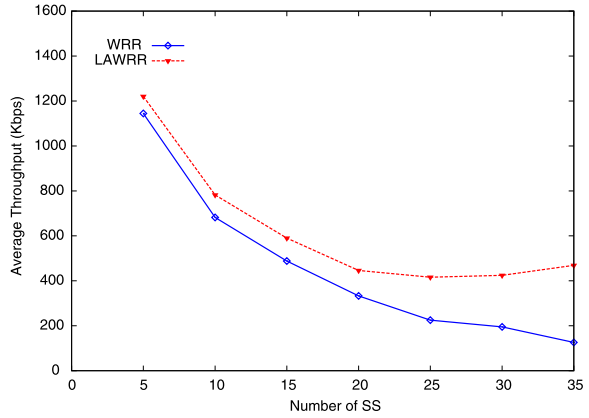

Figure 9 Average throughput per SS for the nrtPS class. It shows that LAWRR outperforms WRR throughout. This is due to the proportionate increase in the number of packets to be transmitted in a round according to the current load characteristics.

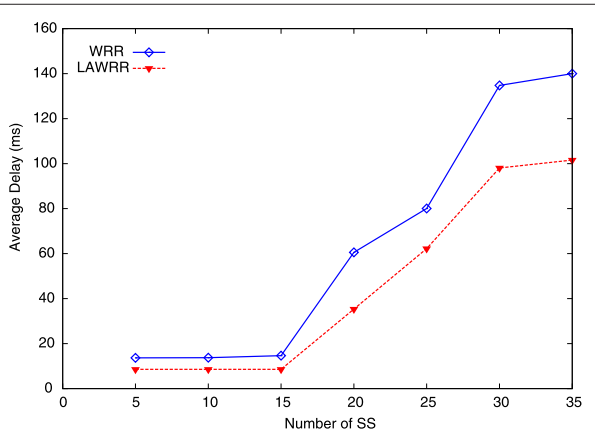

Figure 11 Average delay per SS for the rtPS class. It shows the packet delay for different numbers of connections for WRR and LAWRR for the rtPS class. It also shows that LAWRR outperforms WRR.

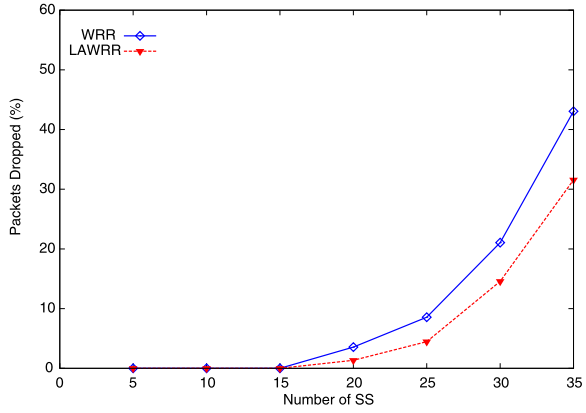

Figure 12 Packet loss per SS for the rtPS class. It shows the packet loss for different numbers of connections for WRR and LAWRR for the rtPS class.

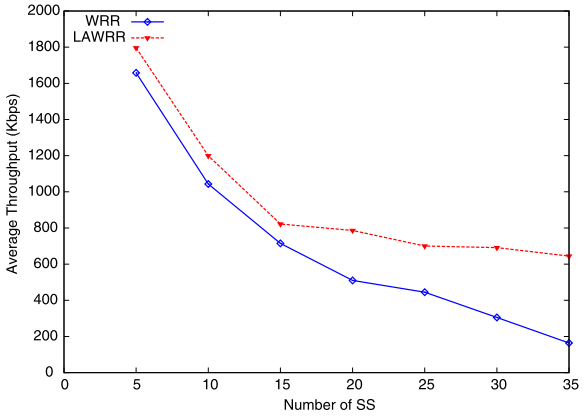

Figure 14 Average throughput per SS for the nrtPS class. It shows that LAWRR performs better than WRR throughout. This is because the number of packets to be transmitted in a round is increased in proportion to the current load characteristics, with a reduction of packet loss.

Figure 15 shows the average throughput for the BE class. LAWRR is better than WRR for 0 to 20 SSs but performs similarly from 20 to 35 SSs because no packets are allowed to be transmitted. The reason for the superior performance of LAWRR here is the opportunity given to the BE class when traffic is light and its service round is proportional to the traffic load characteristics of each queue, which yields an increase in the number of packets transmitted in a round. LAWRR improves the throughput by 35.2% over WRR.

To verify the effectiveness of our proposed algorithm, Figure 16 compares LAWRR and WRR in terms of packet drop ratio versus downlink buffer size. It can be seen that LAWRR drops fewer packets than WRR. The reason is that burstiness of input traffic arrive while LAWRR sustains the specified value of the arrival by dynamically adjusting the weight according to the traffic load characteristics, and hence, the total number of dropped packets using LAWRR is far less than that when using WRR. On the other hand, WRR only allows the serving of a static number; therefore, there are more packets dropped compared with LAWRR.

6 Conclusions

We have presented a new scheduling algorithm, LAWRR, to address the burstiness of input traffic in IEEE 802.16

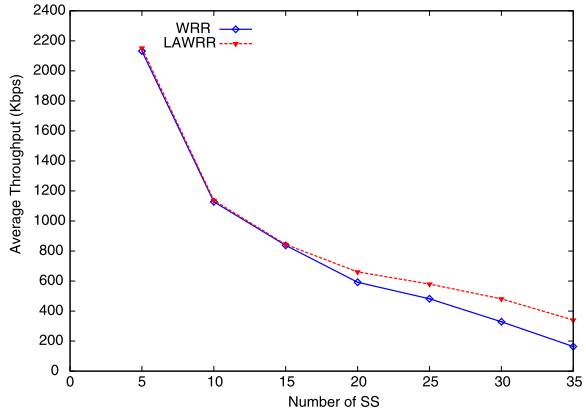

Figure 13 Average throughput per SS for the rtPS class. For 0 to 15 SSs, the algorithms are similar because there is no packets loss, given the low traffic load. But from 15 to 35 SSs, LAWRR outperforms WRR because it increases the number of packets transmitted in response to bursty traffic.

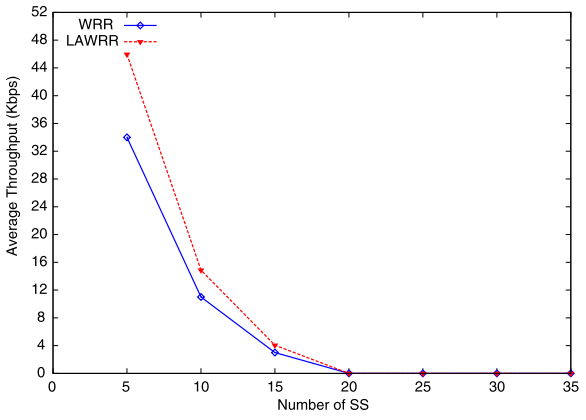

Figure 15 Average throughput per SS for the BE class. LAWRR is better than WRR for 0 to 20 SSs but performs similarly from 20 to 35 SSs because no packets are allowed to be transmitted. The reason for the superior performance of LAWRR here is the opportunity given to the BE class when traffic is light and its service round is proportional to the traffic load characteristics of each queue, which yields an increase in the number of packets transmitted in a round.

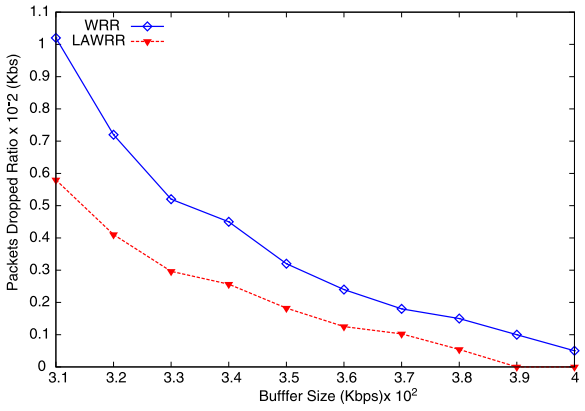

Figure 16 Comparison of packets dropped by LAWRR and WRR versus buffer size.

networks. A number of simulation experiments were conducted to evaluate its performance. The results show that LAWRR significantly outperforms WRR in terms of average delay and average packet loss, as well as in average throughput. Finally, we also investigated the effectiveness of our proposed LAWRR running within conditions of finite buffer size, which again showed LAWRR to outperform WRR.

Competing interests

The authors declare that they have no competing interests.

Acknowledgements

This work was supported by the Research University Grant Scheme (RUGS) at Universiti Putra Malaysia.

Author details

[1]Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, 43400 UPM Serdang, Selangor DE, Malaysia. [2]Sports Academy, Universiti Putra Malaysia, 43400 UPM Serdang, Selangor DE, Malaysia.

Received: 18 December 2013 Accepted: 26 November 2014
Published: 19 December 2014

References

1. IEEE 802.16 Working Group and others, IEEE Standard for Local and Metropolitan Area Networks, Part 16: air interface for fixed broadband wireless access systems. IEEE Std, 802, 16–2004 (2004)
2. JF Borin, NLS da Fonseca, Simulator for WiMAX networks. Simul. Model. Practice Theory. **16**(7), 817–833 (2008)
3. C Wang, Y Wan-Jhen, L Hao-Kai, Dynamic admission control and bandwidth reservation for IEEE 802.16 e mobile WiMAX networks. EURASIP J. Wireless Commun. Netw. **1**, 1–20 (2012)
4. TM Nguyen, Y Taihyung, J Youchan, K Yeunwoong, P Jinwoo, QoS-aware dynamic resource allocation for wireless broadband access networks. EURASIP J. Wireless Commun. Netw. **1**, 1–12 (2014)
5. Y-W Chen, Y-Y Chu, I-H Peng, Energy-saving centric uplink scheduling scheme for broadband wireless access networks. EURASIP J. Wireless Commun. Netw. **1**(70) (2014)
6. Y Cao, VK Li, Scheduling algorithms in broadband wireless networks. Proc. IEEE. **89**, 76–87 (2001)
7. C Cicconetti, L Lenzini, E Mingozzi, C Eklund, Quality of service support in IEEE 802.16 networks. Netw. IEEE. **20**(2), 50–55 (2006)
8. A Sayenko, O Alanen, T Hämälinen, Scheduling solution for the IEEE 802.16 base station. Comput. Netw. **52**(1), 96–115 (2008)
9. C-P Lin, J Chen, H-L Chen, An efficient bandwidth allocation algorithm for real-time VBR stream transmission under IEEE 802.16 wireless networks. J. Netw. Comput. Appl. **33**(4), 467–476 (2010)
10. W Nie, H Wang, JH Park, Packet scheduling with QoS and fairness for downlink traffic in WiMAX Networks. JIPS. **7**(2), 261–270 (2011)
11. W Mardini, MA Alfool, Modified WRR scheduling algorithm for WiMAX networks. Netw. Protoc. Algorithms. **3**(2), 24–53 (2011)
12. H Shimonishi, M Yoshida, R Fan, H Suzuki, in *Global Telecommunications Conference, 1997. GLOBECOM'97.*, IEEE vol. 2. An improvement of weighted round robin cell scheduling in ATM networks (*IEEE*, 1997), pp. 1119–1123
13. H-H Changhwan, HK Yoon, OK Kim, in *Proceedings of the IEEE Region 10 Conference*. 1999, a queue length-based scheduling scheme in ATM networks, TENCON 99, vol. 1 (IEEE, 1999), pp. 234–237
14. M Katevenis, S Sidiropoulos, C Courcoubetis, Weighted round-robin cell multiplexing in a general-purpose ATM switch chip. Selected Areas Commun. IEEE J. **9**(8), 1265–1279 (1991)
15. MA Teixeira, PR Guardieiro, A new and efficient adaptive scheduling packets for the uplink traffic in WiMAX networks. EURASIP J. Wireless Commun. Netw. **1**, 1–11 (2011)
16. C Semeria, Supporting differentiated service classes: queue scheduling disciplines. White Paper, Part Number:200020-001 12/01 Juniper Netw, 1-27 (2001)
17. EL Hahne, Round robin scheduling for fair flow control in data communication networks. Ph.D. thesis, Dept. Elect. Eng. and Comput. Sci., M.I.T., 1-222 (December,1986)
18. X Meng, *An efficient scheduling for diverse QoS requirements in WiMAX,* Masters thesis, Department of Electrical and Computer Engineering. (University of Waterloo, Ontario, Canada, 2007)
19. M Sarkar, H Sachdeva, in *Proceedings of IAENG Conference on World Congress on Engineering and Computer Science, Berkeley*. A QoS aware packet scheduling scheme for WiMAX (1 Berkeley California, USA, 2009)
20. NA Ali, P Dhrona, H Hassanein, A performance study of uplink scheduling algorithms in point-to-multipoint WiMAX networks. Comput. Commun. **32**(3), 511–521 (2009)
21. MAS Khan, AR Sattar, T Mustafa, S Ahmad, Performance evaluation and enhancement of uplink scheduling algorithms in point to multipoint WiMAX networks. European J. Sci. Res. (EuroJournals). **42**, 491–506 (2010)
22. M Gidlund, G Wang, Uplink scheduling algorithms for QoS support in broadband wireless access networks. J. Commun. **4**(2), 133–142 (2009)
23. K Mezger, DW Petr, Bounded delay for weighted round robin. Telecommunications and Information Sciences Laboratory, Department of Electrical Engineering and Computer Sciences, University of Kansas Technical Report -10230-07 1–34 (1995)
24. M-el-A Brahmia, A Abdelhafid, P Lorenz, Adaptive scheduling mechanism for IPTV over WiMAX IEEE 802.16 j networks. Int. J. Commun. Syst. **27**(7), 1009–1019 (2012)
25. Y Ito, S Tasaka, Y Ishibashi, in *Performance, Computing, and Communications Conference, 2002. 21st IEEE International*. Variably weighted round robin queueing for core IP routers (IEEE, 2002), pp. 159–166
26. C Cicconetti, A Erta, L Lenzini, E Mingozzi, Performance evaluation of the IEEE 802.16 MAC for QoS support. Mobile Comput. IEEE Trans. **6**(1), 26–38 (2007)
27. H Wang, L Dittmann, Downlink resource management for QoS scheduling in IEEE 802.16 WiMAX networks. Comput. Commun. **33**(8), 940–953 (2010)

doi:10.1186/1687-1499-2014-226
Cite this article as: Saidu *et al.*: **A load-aware weighted round-robin algorithm for IEEE 802.16 networks.** *EURASIP Journal on Wireless Communications and Networking* 2014 **2014**:226.