WIRELESS COMMUNICATIONS AND MOBILE COMPUTING
Wirel. Commun. Mob. Comput. 2015; **15**:584–600
Published online 2 May 2013 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/wcm.2368

RESEARCH ARTICLE

A preemption-based scheduling algorithm for WiMAX networks

Haidar Safa* and Samar Khayat

Department of Computer Science, American University of Beirut, P.O.Box: 11-0236 Riad El-Solh, Beirut 1107-2020, Lebanon

ABSTRACT

Several scheduling techniques were designed for the base station (BS) of IEEE 802.16e wireless interoperability of microwave access networks. However, depending on the BS scheduler alone to determine the servicing order of each connection might affect the accuracy of the scheduling process because the BS does not necessarily have enough up-to-date information about the current state of the connections at the subscriber station. In this paper, we propose a preemption-based scheduling algorithm that focuses on improving the quality of service requirements of real-time service flow classes. The proposed algorithm incorporates two schedulers, one at the BS and another one at the subscriber station. We have implemented and integrated the proposed algorithm with the network simulator NS2 using the Network and Distributed Systems Laboratory wireless interoperability of microwave access module. Simulation results have shown that the proposed approach outperforms other scheduling algorithms in terms of enhancing the throughput and the average delay of real-time quality of service classes. Copyright © 2013 John Wiley & Sons, Ltd.

KEYWORDS
WiMAX; resource allocation; scheduling; QoS; wireless broadband access networks

***Correspondence**
Haidar Safa, Department of Computer Science, American University of Beirut, P.O.Box: 11-0236 Riad El-Solh, Beirut 1107-2020, Lebanon.
E-mail: hs33@aub.edu.lb

1. INTRODUCTION

The IEEE 802.16 standard defines the specifications of the medium access control (MAC) and physical layers for broadband wireless access systems [1,2]. It introduced a cost-effective wireless broadband technology called WiMAX, short for *wireless interoperability of microwave access*. Its MAC layer is designed to handle the communication between a set of mobile/stationary subscriber stations (MSs or SSs) and a base station (BS) while providing high quality of service (QoS) mechanisms to support a large variety of applications. The BS is a central equipment set providing connectivity, management, and control of several SSs situated at varying distances, whereas a subscriber station (SS) can represent a building equipped with conventional wireless or wired LAN. An MS is an SS intended to be used while in motion or during halts at unspecified points. The communication path between an SS and a BS is bidirectional, a downlink (DL; BS to SS) and an uplink (UL; SS to BS). For the UL and DL data transmissions, time is divided into frames of fixed duration. A frame is divided into two subframes, the UL subframe and the DL subframe, that define the features of the physical channels. These two subframes are duplexed using either frequency division duplex or time division duplex. The DL subframe that is periodically generated by the BS contains the DL-MAP and UL-MAP, which are used to announce the information of the arrangement of the DL and UL periods in the frame. The DL-MAP specifies the DL channel access and the associated burst profile. The UL-MAP defines the UL channel access, that is, the time slot in which the SS can transmit in the UL subframe and the UL data burst profiles [2]. The UL-MAP grants bandwidth to specific SSs; then, the SSs will transmit in their assigned allocation. The UL subframe contains the data sent by different SSs.

Three basic modules are necessary in IEEE 802.16 for managing QoS: the admission controller that determines whether a new request for a connection can be granted or not according to the remaining free bandwidth, the scheduler that determines which packet should be served first in such a way to guarantee QoS requirements, and the buffer manager that controls the buffer size and decides which packets to drop upon buffer overload. The IEEE

802.16 standard specifies five classes of applications, each of which has an obligatory set of network-related QoS parameters, traffic specifications and requirements. However, the standard does not specify any mandatory and informative QoS component, which includes scheduling and admission control functions.

Designing a proper scheduling algorithm at the MAC layer that provides flexibility and efficiency over a range of different traffic types and QoS requirements while still achieving high system utilization is a major challenge. Also, depending on the BS scheduler alone to determine the servicing order of each connection might affect the accuracy of the scheduling process and the degree of QoS that can be provided because the BS does not necessarily have enough up-to-date information about the current state of the connections at the SS.

In this paper, we propose a scheduling algorithm that provides more chances to delay sensitive traffic to be serviced while still maintaining fairness for various traffic types. This algorithm, which is an extension to our work proposed in [3], is preemption-based and referred to as SS-assisted scheduling algorithm because it is divided into two schedulers, one runs at the SS and another one at the BS. In this approach, the newly arriving critical real-time packets at the SS and the BS DL will have better chances to meet their deadlines. Moreover, it improves bandwidth utilization by dynamically dividing it between the UL and DL subframes. The remainder of this paper is organized as follows. In Section 2, we briefly describe the background of WiMAX networks and survey related work. In Section 3, we introduce the proposed SS-assisted scheduling algorithm and present its design and implementation details. Section 4 evaluates the performance of the proposed algorithm, compares it with other approaches, and analyzes the obtained results. Finally, we conclude in Section 5.

2. BACKGROUND AND RELATED WORK

2.1. Background

The IEEE 802.16 MAC layer is a connection-oriented layer that comprises three sublayers: the service-specific convergence sublayer, the common part sublayer, and the security sublayer (SS). The convergence sublayer accepts protocol data units from the higher layers, transforms them into MAC service data units (SDUs), and passes them to the common part sublayer where the core MAC functionalities such as system access, bandwidth allocation, connection establishment, and scheduling are provided. The security sublayer provides authentication, secure key exchange, and encryption.

In WiMAX, an SS registers to the BS first (i.e., the initial ranging process) using its 48-bit universal MAC address that uniquely defines its air interface. The BS is the only transmitter operating in the DL direction. All SSs receive the same transmission and retain only those protocol data units addressed to them. On the other hand, SSs share the UL to the BS on a demand basis. Each connection between the SS and the BS is identified with a unique 16-bit connection identifier (CID), and each packet has to be associated with a connection. Shortly after SS registration, connections are associated with service flows to provide a reference against which to request bandwidth. A service flow is a unidirectional flow of packets and is identified by a service flow identifier. It is characterized by a set of QoS parameters that quantify aspects of its behavior, known as a QoS class. Main QoS parameters are the traffic priority that represents the priority assigned to a service flow, the maximum sustained traffic rate that represents the peak information rate of the service, the maximum traffic burst that represents the maximum continuous burst the system should accommodate for the service, the minimum reserved traffic rate that represents the minimum amount of data to be transported when averaged over time, the tolerated jitter that represents the maximum delay variation for the connection and the maximum latency (ML) that represents the maximum allowable time between the reception of a packet on the network interface and forwarding it to its radio frequency interface.

When a service flow is admitted or active, it is mapped to a connection that can handle its QoS requirements and then forwarded to the appropriate queue according to its service class as shown in Figure 1, which represents the Network and Distributed Systems Laboratory (NDSL) implementation of the Point-to-Multipoint WiMAX module [4]. The SS sends next a bandwidth request message reporting the current queue size of each of its connection to the BS UL bandwidth allocation scheduler. As a result of receiving bandwidth request messages, the BS grants bandwidth to each SS using the UL-MAP that is transmitted by the BS at the beginning of each DL subframe. Five QoS classes were defined in the 802.16 [1,2]:

(1) *The Unsolicited Grant Service* (UGS) is designed to support real-time UL service flows that transport fixed-size data packets on a periodic basis such as voice over Internet Protocol.
(2) *The Real-Time Polling Service* (rtPS) is designed to support real-time UL service flows that transport variable-sized data packets on a periodic basis, such as MPEG video.
(3) *The Non-Real-Time Polling Service* (nrtPS) is designed to support delay-tolerant data streams consisting of variable-sized data packets for which a minimum data rate is required, such as FTP.
(4) *The Best Effort* (BE) is designed to support data streams for which no minimum service level is required (i.e., best effort traffic in the UL).
(5) *The Extended Real-Time Polling Service* (ertPS) is built on the efficiency of both UGS and rtPS and designed to support real-time service flows that generate variable-sized data packets on a periodic basis.

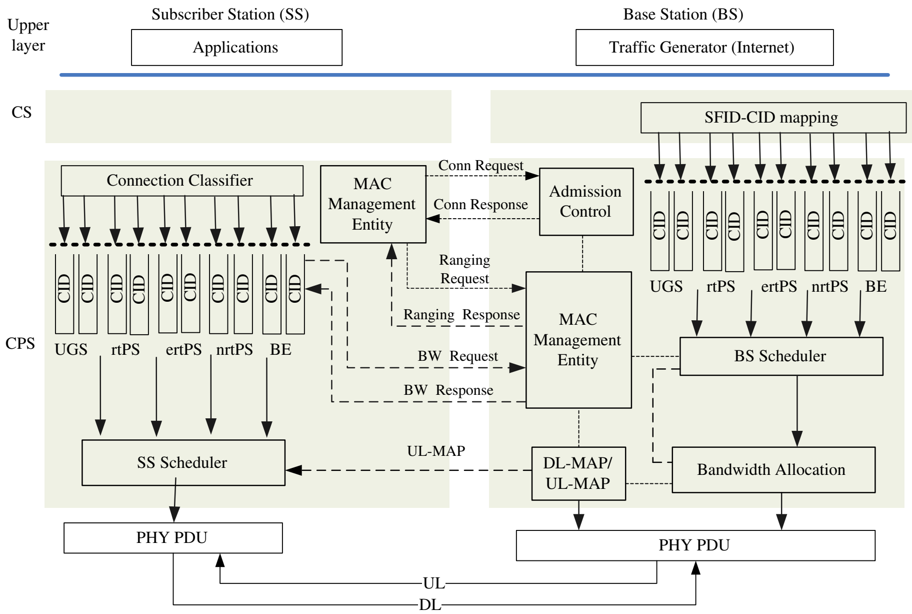


Figure 1. The network and distributed systems laboratory medium access control (MAC) architecture of IEEE 802.16.

By specifying a scheduling type and its associated QoS parameters, the BS scheduler can anticipate the throughput and latency needs of the UL traffic and provide polls and/or grants at the appropriate time.

2.2. Related work

Several scheduling algorithms were proposed for WiMAX networks [3–13]. These algorithms aim at allocating resources to users while meeting QoS requirements for real-time transmission with the delay and delay jitter constraints.

The round robin (RR) scheduling algorithm fairly assigns the allocation one by one to all connections [14]. The traditional deficit round robin (DRR) that was proposed in [11] is packet-based scheduling discipline widely used in wired networks where link capacities do not change with time. This traditional DRR cannot be directly used in WiMAX because, in 802.16e networks, the link capacity can change over time, and it also depends on the user location. Hence, a variation of DRR with fragmentation (DRRF) was proposed in [12] to allocate resources in WiMAX networks while allowing for varying link capacity. In RR, the allocation may be made for connections that have nothing to transmit. Therefore, some modifications need to be made to skip the idle connections and allocate only to active connections. Because RR cannot assure QoS for different service classes, RR with weight, weighted round robin (WRR), has been applied for WiMAX scheduling [4,7,10,15]. The weights are in terms of queue length and packet delay and can be used to adjust for the throughput and delay requirements.

A deficit fair priority queue (DFPQ)-based scheduler was proposed in [6] in which six queues are defined according to their direction (UL and DL) and the service classes (rtPS, nrtPS, and BE). In this approach, DFPQ is used to distribute the total available bandwidth among service flows. Earliest deadline first (EDF) [16] is used for the rtPS queues, weighted fair queuing [17] is used for the nrtPS queues, and RR [14] is used to allocate the remaining bandwidth for the BE queues. In each round in DFPQ, the highest priority queue will be served first until its assigned bandwidth is deficit after which lower queues will have the chance to be served. The queues are ordered according to the service classes (rtPS > nrtPS > BE), and within each service class, the DL transmission has a higher priority than the UL transmission. Similar to [14], the DFPQ algorithm assigns a *quantum* (Qt) for each queue that represents the maximum number of bits that can be serviced in the first round. In each round, the scheduler visits nonempty queues according to their order. Each queue is allowed to service up to Qt bits in the first round. If more packets still exist in the queue after servicing Qt bits, the remaining amount of bits is stored in a variable called the *deficit counter* (DC). In subsequent rounds, the new Qt of each flow is the sum of the DC from the previous round and its Qt. In DFPQ, the scheduler might interrupt servicing the rtPS queue and start servicing the BE queue even if there are still some packets in the rtPS queue. As a result, these packets may expire if their deadlines expire or if there is no more available bandwidth.

To solve this issue, we proposed a preemptive deficit fair priority queuing (PDFPQ) in [9]. PDFPQ maintained the same scheduling queues. However, the notion of preemption was added to the architecture to improve the performance of the rtPS service flow packets by giving them higher chances to meet their deadlines while decreasing their delay to guarantee the QoS requirements. The main limitation of PDFPQ algorithm is that it assumes real-time scheduling can occur at the BS for the UL subframe. However, in reality, the BS UL scheduler does not receive requests while scheduling; all packet arrivals occur at the beginning of each frame, which makes preemption not feasible for the UL scheduling at the BS. Furthermore, ertPS traffic is not considered among the real-time service flows in both DFPQ and PDFPQ.

The scheduling algorithm that was proposed in [10] is capable of allocating slots on the basis of the QoS requirements, bandwidth request sizes, and the network parameters. It ensures the QoS requirements for all the service classes and fairly shares free resources. It proposes for each service class, a policy to allocate slots. It comprises three major stages. First, the BS calculates the minimum number of slots to be allocated for each connection to ensure the basic QoS requirements. Second, the BS assigns free unused slots to some connections. Third, it selects the order of slots in a way to improve the provisioning of the QoS guarantees. Although in this proposal, there is a possibility to assign explicitly each slot to some contention thus specifying the serving order, it makes scheduling in WiMAX more complicated.

These scheduling algorithms and many others were designed for the BS. However, depending on the BS scheduler alone to determine the servicing order of each connection might affect the accuracy of the scheduling process and the degree of QoS because the BS may not be aware of the current state of the connections at the SS. Moreover, any SS may have several simultaneous active connections each of which has different QoS requirements and cannot be scheduled in a common first in, first out queue at the SS scheduler. Also, to reduce its workload, the BS UL scheduler does not require detailed information on every connection at the SS and gives one grant for all the connections rather than grant per connection. Thus, when an SS is assigned an UL grant, it is non-deterministic for which of its connections it was intended, and consequently, a scheduler at the SS MAC has to allocate the granted bandwidth to some or all of its connections. Employing a local scheduler in each SS to allocate the UL slots granted by the BS to their respective connections can help urgent critical packets arriving at the SS to overtake other less urgent packets.

3. THE PROPOSED SS-ASSISTED SCHEDULING ARCHITECTURE

In this section, we present our new *subscriber station (SS)-assisted scheduling algorithm* which mainly focuses on improving QoS requirements of the delay and throughput of real-time service flow classes.

3.1. Basic idea

The proposed algorithm is hierarchical in which eight queues are defined according to their direction (UL and DL) and the service classes (ertPS, rtPS, nrtPS, and BE) as shown in Figure 2. The first layer contains a scheduler that distributes dynamically the total bandwidth among service flows in the eight queues and gives real-time packets more chances to satisfy their QoS requirements. The eight queues are classified as either preemptive or non-preemptive queues. Indeed, queues of the rtPS and ertPS service flows in both the UL and DL direction are non-preemptive queues, whereas other queues can be preempted under certain conditions. The proposed algorithm is divided into two sub-algorithms, a scheduler at the BS and another one at the SS, where preemption is applied to both of them. At the SS, a preemptive queue being serviced can be preempted if a packet, which cannot be delivered within its deadline, arrives to a non-preemptive queue. However, at the BS, a preemptive queue being serviced is preempted only if a critical packet arrives to a non-preemptive DL queue. In this case, the newly arriving packet will be serviced first, and the preempted queue will continue to be serviced afterwards.

An rtPS or ertPS packet is defined as critical if its deadline expires before the next frame is totally served and the remaining capacity of the current frame is not enough to service this packet. More formally, an rtPS or ertPS data grant k is said to be critical if (1) and (2) are satisfied:

$$deadline_k < (frame_n + 1) \times frameDuration \quad (1)$$

$$size_k > C_a - DC_i \quad (2)$$

where $frame_n$ is the current frame number, C_a is the available capacity in the current frame, $size_k$ is the size of the current packet k, and DC_i is the deficit counter of the ith service flow belonging to the preemptive queue, which will be preempted or is about to be serviced.

For each non-preemptive queue, a *quantum critical* (Q_{crit}) is defined to give the queue another chance to

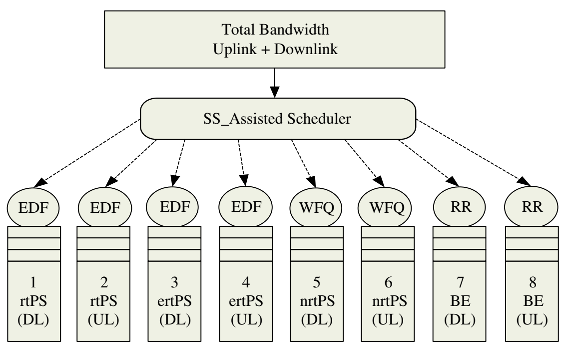


Figure 2. The hierarchy of proposed algorithm.

service critical packets. The Q_{crit} value is a percentage of the original value of the queue's quantum. Queues are allowed to use Q_{crit} to serve critical packets only. The processing of critical packets continues until the Q_{crit} of the non-preemptive queue becomes less than or equal to zero. Q_{crit} is initialized only once per frame and not at every round.

In the proposed approach, a connection is admitted only when its minimum traffic rate is zero (BE connections) or when the estimated value of the remaining available bandwidth is positive. The available bandwidth, B_a, could be estimated by deducting the total minimum traffic rates, R_{min}, of all connections of all the service classes from the total bandwidth B_{total}. For example, assume that there are n service classes and the i^{th} class has totally J_i connections, then the available bandwidth can be given as

$$B_a = B_{total} - \sum_{i=0}^{n} \sum_{j=0}^{J_i-1} R_{\min}(i,j) \tag{3}$$

where $R_{min}(i,j)$ is the minimum reserved traffic rate of the j^{th} connection in the i^{th} class of service flow and B_{total} is the total bandwidth.

For every admitted and active real-time service flow, some parameters are maintained and are used to approximate the expected delay of each real-time connection. The *reference time*, t_0, is used to calculate both the generation time and the deadline of the real-time data grants (i.e., it is the creation time of the service flow to which a data grant belongs). For each service flow, there is the *interval* in which a specific number of rtPS or ertPS grants is generated. Therefore, the expected *generation time,* t_k, of each data grant k in a service flow is estimated as [8]

$$t_k = t_0 + (k \times interval) \tag{4}$$

Moreover, each datagram can be marked with an approximate delivery deadline that is used to determine whether the packet is critical or not:

$$deadline_k = t_k + jitter \tag{5}$$

where *jitter* is the maximum delay variation that must be enforced for each service flow.

The operational flow of the proposed algorithm can be summarized as follows. The SSs send for each connection a message to the BS indicating the overall amount of requested bandwidth. At the BS, these bandwidth requests are grouped into UL queues according to their service flow type. Similarly, the DL packets at the BS are grouped into DL queues. The UGS queues have the highest priority among all the queues and will be served first. The proposed scheduling algorithm is applied at the BS (i.e., BS-Scheduler) for the remaining queues according to the following priority order (DL-rtPS > UL-rtPS > DL-ertPS > UL-ertPS > DL-nrtPS > UL-nrtPS > DL-BE > UL-BE). The preemption concept is applied at the BS only when critical packets arrive to the DL queues. The BS groups the UL grants addressed to the same SS in MAPs. Upon receiving its grants through the MAPs, the SS shares the total bandwidth granted to it among its connections. The proposed scheduler is applied at the SS (i.e., SS-Scheduler) on the non-UGS service flow queues (rtPS > ertPS > nrtpS > BE). Hence, in this approach, critical real-time packets, especially the newly arriving ones at the SS and the BS DL, will have better chances to meet their deadline. Moreover, bandwidth utilization is improved because the bandwidth is dynamically divided between the UL and DL subframes, and fairness is still maintained at both BS and SS schedulers because lower priority queues will eventually be serviced at both schedulers.

3.2. Design and class diagram of the proposed algorithm

Four main components were designed to implement the proposed approach. They are as follows: the admission control, the BS bandwidth management, the BS DL scheduler, and the SS UL scheduler. The major part of this architecture is the scheduling algorithm, which is modeled in two classes the *SSAssisted_BS* and the *SSAssisted_SS* classes as shown in Figure 3. The SSAssisted_BS class represents the BS DL scheduler. It keeps information about the frame, the service queues, the remaining bandwidth, and the resulting frame allocation. It also maintains functions mainly for admitting connections, scheduling packets

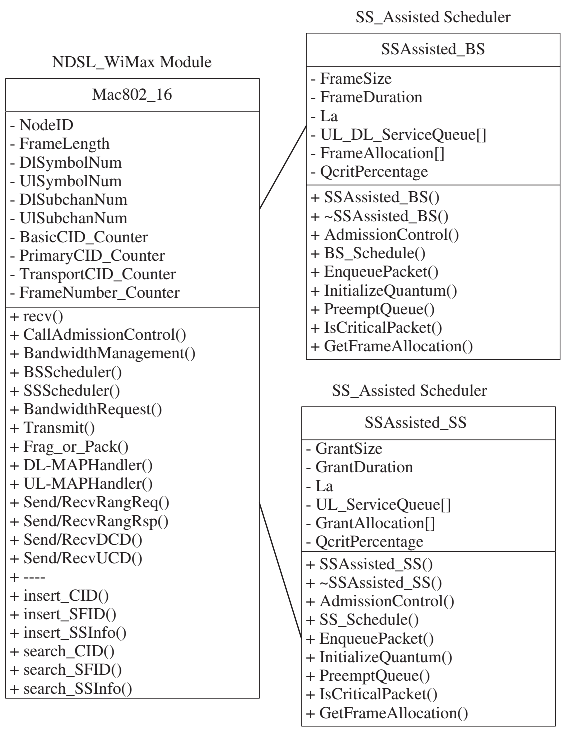


Figure 3. A simplified class diagram of the proposed subscriber station (SS)-assisted scheduling algorithm.

at the BS, preempting queues, determining critical packets, and forming the final frame allocation. On the other hand, the SSAssisted_SS class represents the SS UL scheduler. It keeps information related to its allocated grant rather than the frame.

In the proposed approach, a (*Qt*) whose value corresponds to the maximum sustained traffic rate, R_{max}, of the scheduling service is maintained for each queue, q. For example, suppose that there are J_i connections for i^{th} class of service flow. The *Qt* for this i^{th} class can be calculated as

$$QT[i] = \sum_{j=0}^{J_i} R_{\max}(i, j) \tag{6}$$

In case if $R_{max} = 0$ (BE service flow), R_{min} will be used instead.

Each queue, q, keeps a *deficit counter* (DC_q) value that is initialized to its quantum value at the beginning of each round. The deficit counter of a queue is decremented by the size of the packet being scheduled; thus, packets in each queue continue to be scheduled until the deficit counter of the queue is either no longer positive or no remaining bandwidth left in the frame. At any time, a critical packet to be queued in a non-preemptive queue, namely an rtPS or an ertPS queue, might arrive at a node while the preemptive queue is being serviced. In such a case, the latter queue will be preempted thus transforming the flow of control to the non-preemptive queue to schedule the critical packets. After servicing these critical packets, the flow of control moves back to continue servicing the preempted queue. Figure 4 summarizes the flow of control of the proposed scheduling approach. To describe this figure, we define C_{total} to be the total capacity of the frame in bytes and C_a to be its remaining capacity. For each service queue q, we initialize the value of C_a to C_{total} and the value of DC_q to QT_q. Also, the initial value of Q_{crit} of queue q is set to a percentage of the original value of the queue's quantum. The algorithm starts then servicing the connections in the service flow with the highest priority queue one by one. Every time a request or a fragment of a request is serviced, DC_q and C_a are decremented by the number of bits that has been serviced. If the queue q that is being serviced is preemptive and a critical packet arrives to one of the non-preemptive queue, then the queue q is preempted, and the flow of control moves to service the critical packets then returns back to q. If the queue q that is being serviced is non-preemptive queue and q has exhausted its DC value (i.e., $DC_q \leqslant 0$) but still contains one or more critical packets, then it uses its Q_{crit} to service the critical packets unless there is no remaining bandwidth in the current frame. Every time Q_{crit} is used, it will be decremented along with C_a by the number of bits that has been serviced. In every scheduling round, we keep iterating over all the queues until C_a becomes negative. This approach permits dynamic distribution of the bandwidth being scheduled and also defines the order and time at which data packets should be sent out through links to their destination nodes.

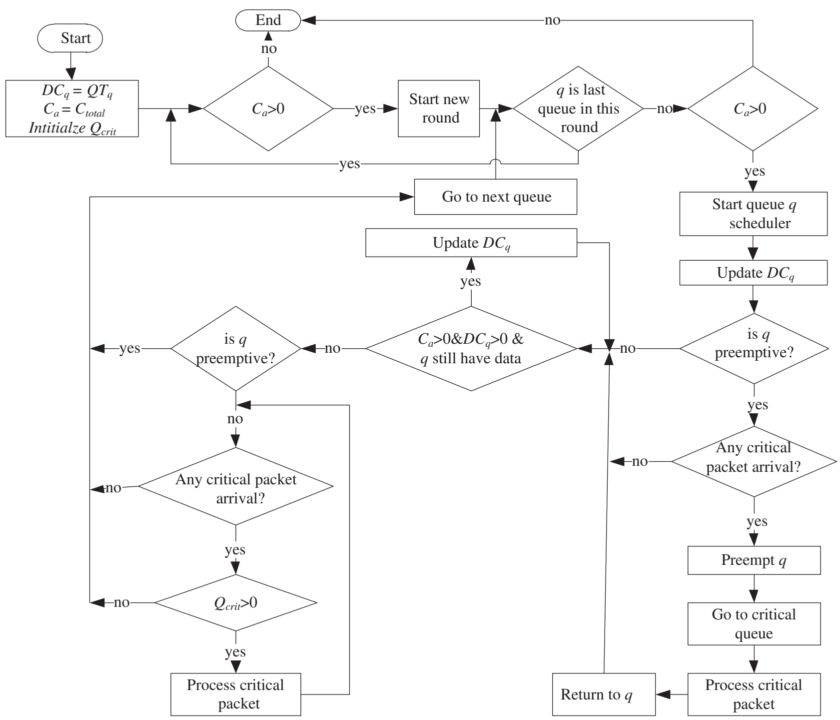


Figure 4. A flow chart of the scheduling algorithm.

3.3. Integrating the proposed approach in Network and Distributed Systems Laboratory

We have incorporated the propose approach in the NDSL module [4] as shown in Figure 5. To do so, some components were added or replaced in addition to updating the MAC802.16 class to adapt it to the proposed algorithm. Moreover, two other classes, the *SSAssisted_BS* and the *SSAssisted_SS* classes, were added to represent the SS-assisted scheduling algorithm at each of the BS and SS as shown later on in Figure 7 and Figure 8.

The NDSL implementation statically divides the frame between the DL and UL subframes by a ratio of 3:1 to distribute the bandwidth among the DL and UL service flows at the BS. However, in our implementation, we dynamically divided the frame between the DL subframe and the UL subframe; then, the scheduling module is used to dynamically distribute the bandwidth between the BS DL and UL service flows. We first schedule the DL data packets from the BS and the bandwidth request packets from the SSs giving priority to critical packets of the BS DL. We create the DL and UL maps. During this scheduling process, BS applies preemption only to the DL queues thus handling the BS critical data packets. The output of this process is the *bandwidth allocation*, which defines the order in which data packets should be sent out through the DL to the SSs and the grants assigned to each SS; thus, the SS can use these grants to schedule its UL data packets.

The DL and UL *MAPS* are shown in Figure 6 and consist mainly of the information elements (IEs), which can be described as follows:

- IEs for UGS data grants are first created in the DL-MAP. After allocating UGS slots, all remaining free slots are then allocated for ertPS, rtPS, nrtPS, and BE while ensuring that critical packets will be serviced within their deadlines. For each DL packet, an IE is created with the CID of the corresponding connection. The number of slots allocated is equivalent to the packet length. At this stage, bandwidth capacity is granted for non-UGS service flows for all SSs.
- An initial ranging IE is created in the UL-MAP.
- For each UL service flow at the BS, a bandwidth request IE is created in the UL-MAP to give each SS a chance to send its bandwidth requests per connection and not per SS.
- For each SS, the total length of non-UGS UL packets is calculated. Then, a data grant IE is created in the UL-MAP to allocate a number of slots equivalent to the sum of the non-UGS requested bandwidth values.

The *BS DL scheduler* is responsible for servicing the DL packets at the BS according to the DL-MAP, which it receives as an input. The DL MAP contains the IEs for each SS. The order of IEs defines the order by which the DL packets should be serviced to better guarantees QoS for real-time packets. For each IE, the bandwidth size is calculated from the number of slots allocated to it and is

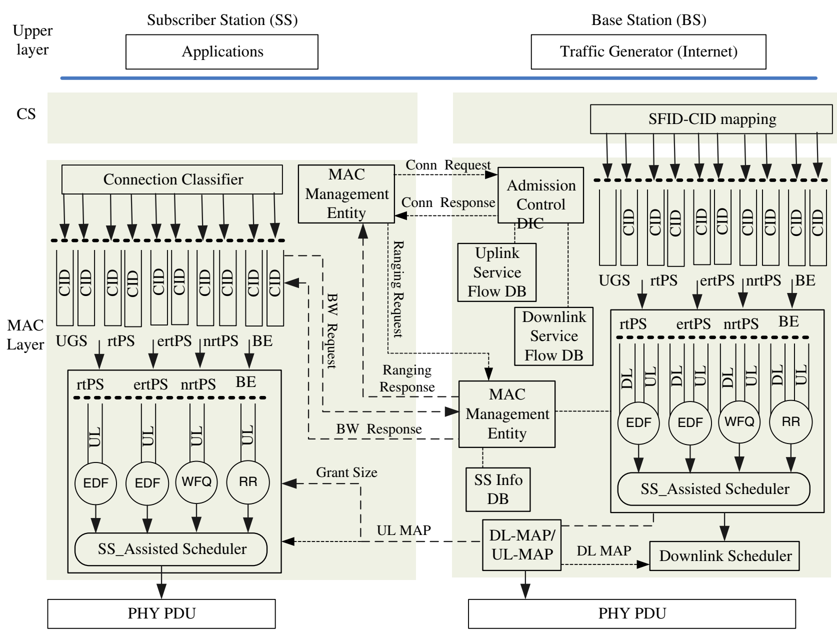


Figure 5. The subscriber station (SS)-assisted scheduling architecture integrated into network and distributed systems laboratory medium access control (MAC).

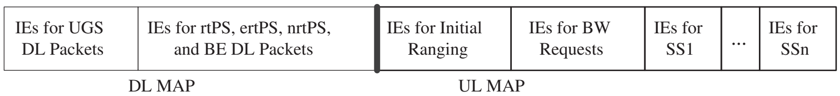


Figure 6. Information elements for the downlink-MAP (DL-MAP) and uplink-MAP (UL-MAP).

used to service DL packets from the BS queues as shown in Figure 7. As a result, UGS packets will all be serviced such that they do not exceed the R_{max} for their connections. Non-UGS packets will be serviced according to their predefined order.

The *SS UL scheduler* component is responsible for scheduling UL data packets according to the UL-MAP, which it takes as an input. It first services all the UGS packets in the UL UGS queue then locates the IE in the MAP, which is assigned the primary CID of the SS. From this IE, it gets the total grant assigned to the SS. When the SS becomes aware of its assigned bandwidth, its UL scheduler starts scheduling the non-UGS packets in its UL queues, applying preemption when needed, as shown in Figure 8. The output of this module defines the order and time at which every data packet should be sent out to the BS.

3.4. Queueing model and delay analysis

The UL channel in WiMAX can be considered as a multiple access communication channel shared by N SSs and can be modeled as multi-class priority queuing system. Each subframe consists of M time slots each of which is normalized to be of unit length τ. In the time division duplex scheme, each SS i may be allocated n_i time slots per frame. Packets of the same priority class are served on a first-come-first-served basis if they arrive at different slots. Otherwise, they are served randomly. At each station, the packet arrival rate is Poisson arrival process where λ_j is the packet average arrival rate of class j (*j*=*1,2,..,J*). A packet may be transmitted in different number of time slots depending on its length. Let B_k^j denote the number of time slots to transmit the kth arriving packet of priority class j. $\{B_k^j, k \geqslant 1\}$ is assumed to be a sequence of independent and identically distributed (i.i.d) random variable for each class j. The transmission requests can be expressed by integer multiples of time slot.

The delay of the kth packet of priority class j is defined as the total time spent by the packet to get through the UL channel and is given as

$$D_k^j = W_k^j + \sigma_k^i + P_k^j + F_L \tag{7}$$

where W_k^j denotes the packet waiting time, σ_k^j denotes the time taken to transmit the kth packet belonging to priority class j, F_L denotes the packet average latency, and P_k^j denotes the packet propagation delay.

The average packet delay of priority class j is given by [18] as

$$\overline{D}_j = W_{j,1} + \left\{ E\left[B_k^j\right] - \frac{M-1}{M} \right\} \left\lfloor \frac{M}{n_i} \right\rfloor \tau + F_L + P^j \tag{8}$$

where

$$W_{j,1} = \frac{\sum_{i=1}^{J} \lambda_i b_{i,2} + (1-\eta_J) \lfloor M/n_i \rfloor \tau}{2(1-\eta_j)(1-\eta_{j-1})} \tag{9}$$

$\lfloor M/n_i \rfloor$ is the biggest integer not larger than M/n_i.

$b_{j,2}$ is the second moment of service time and is given by

$$\begin{aligned} b_{j,2} &= \int_0^\infty x^2 dB_j(x) \\ \rho_j &= \lambda_j b_{j,1} \\ \eta_j &= \sum_{i=1}^{j} \rho_i \qquad j = 1, 2, ..., J \end{aligned} \tag{10}$$

We can set *J*=1, *J*=2, *J*=3, *J*=4, *J*=5 for UGS, rtPS , ertPS, nrtPS, and BE traffic, respectively.

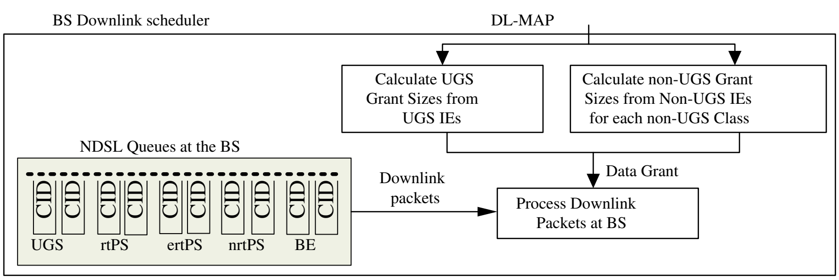


Figure 7. The base station (BS) downlink scheduler.

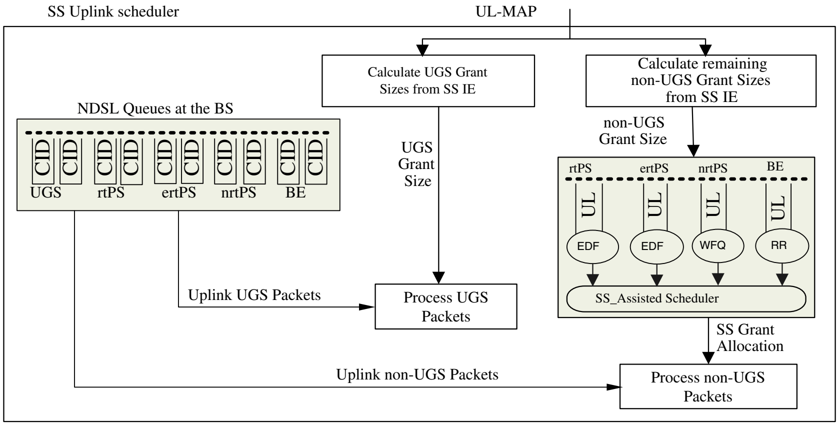


Figure 8. The subscriber station (SS) uplink scheduler.

4. SIMULATION AND RESULT ANALYSIS

4.1. Simulation environment

The proposed SS-assisted scheduling algorithm was simulated under the Network Simulator NS-2 [19] using first a network topology consisting of 20 SS nodes interconnected through a single BS node. Then, the number of SSs was varied to test the scalability of the proposed approach. The NDSL implementation of WiMAX module has five types of traffic flow sources (i.e., UGS, rtPS, ertPS, nrtPS, and BE) [4]. Similarly, in these simulation experiments, each SS node in the simulation model is loaded with five traffic sources each of which generates a specific service flow type. These flows are all generated from the NDSL traffic generator agent in both the SSs and the BS. As a result, each SS maintains five connections, one for each service flow type. The SSs exchange packets of different scheduling services. The BS maintains one connection from each service flow in the DL and one connection from each service flow for each of the SSs in the UL.

4.2. Assumptions and measurement parameters

In this simulation, the following assumptions were made:

- We assume ideal channel conditions, that is, no packet corruption occurs because of any wireless channel impairment.
- All bandwidth request arrivals at the BS occur at the beginning of each frame.
- We analyze the system while it is in a steady state.
- We have used values for QoS parameters similar to the ones used within the NDSL WiMAX module. They are shown per service flow type in Table I.
- We assume a total bandwidth of 20 Mbps and frame duration of 10 ms. The simulation parameters are given in Table II. Some of these parameters were inspired from [15,20].

4.3. Result analysis

We have assessed the performance and compared it with WRR algorithm, which was applied for WiMAX scheduling in [4,7,10,15], and with DFPQ algorithm [6] and PDFPQ [9]. Also, these algorithms incorporate most of the scheduling algorithms that were evaluated in [5]. The number of runs required for each simulation scenario was estimated using the central limit theorem [21], which indicated that 10 runs should be carried out for every experiment to obtain 90 percent confidence interval with a maximum precision level value of 4. The duration of each simulation was set to 300 s. In this performance evaluation, we have measured the following metrics:

Table I. Service flows QoS parameters.

	UGS	rtPS	ertPS	nrtPS	BE
Maximum traffic rate	64 kbps	1024 kbps	64 kbps	1024 kbps	1024 kbps
Minimum traffic rate	64 kbps	512 kbps	8 kbps	512 kbps	—
Maximum latency	50 ms	50 ms	50 ms	500 ms	1000 ms
Tolerated jitter	10 ms	—	30 ms	—	—

UGS, unsolicited grant service; rtPS, real-time polling service; ertPS, extended real-time polling service; nrtPS, non-real-time polling service; BE, best effort.

Table II. Simulation parameters.

Parameter		Value
Channel bandwidth		20 MHz
Frame duration		10 ms
Total OFDMA Symbols Per Frame		48
Total subchannels per frame		152
Ranging opp. per frame		12 OFDMA Symbols
Max. no. of ranging retry		10
Bandwidth req. opp. per frame		12 OFDMA Symbols
Max. no. of bandwidth req. retry		10
DL/UL ratio	WRR	3:1
	DFPQ/SS-Assisted	Dynamic
Modulation		QPSK, 16-QAM, 64-QAM

OFDMA, orthogonal frequency-division multiple access; DL, downlink; UL, uplink; WRR, weighted round robin; DFPQ, deficit fair priority queue; SS, subscriber station; QPSK, quadrature phase shift keying.

(1) End-to-end delay: defined as the time interval between the arrival of an SDU at the MAC layer of the source node (SS/BS) and the complete delivery of this SDU to the next protocol layer at the destination node (BS/SS).
(2) Throughput: defined as the overall amount of user data carried out by the system per unit time (i.e., measured by kbps).
(3) Fairness: defined as the boundedness of the throughput difference between services having different priorities

4.3.1. Average end-to-end delay analysis.

In this section, we study the average end-to-end delay for real-time rtPS and ertPS QoS classes in the three algorithms. Equation (11) calculates this delay per frame j for each service flow i.

$$AverageDelay_i = \left(\frac{\sum_{j=t_n}^{j=t_{n+1}} deliveryTime - \sum_{j=t_n}^{j=t_{n+1}} arrivalTime}{\sum_{j=t_n}^{j=t_{n+1}} receivedPacket} \right) \tag{11}$$

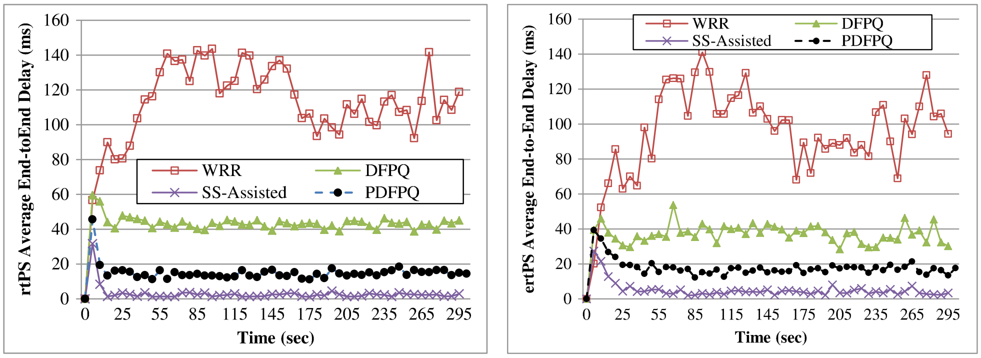
where *arrivalTime* is the time at which a packet arrives at the MAC layer of the source node, *deliveryTime* is the time at which the packet is delivered to the MAC layer at the destination node, and *receivedPacket* is the total number of packets received between the start time, t_n, of frame n and the start time, t_{n+1}, of the next frame $n+1$.

Figure 9 shows the rtPS and ertPS average end-to-end delay for the four scheduling algorithms: the WRR algorithm, the DFPQ algorithm, the PDFPQ algorithm, and the proposed SS-assisted algorithm. We observe that, in WRR, the delay for these two real-time classes increases with time until exceeding the maximum delay limitation as required by their QoS latency parameters. This because the WRR scheduler does not suppress delay for rtPS and ertPS classes, which is a QoS requirement for real-time applications. We also observe that, in DFPQ, PDFPQ, and SS-assisted algorithms, the average end-to-end delay for real-time packets increases at the beginning of the simulation time to its maximum value and then decreases to become almost stable. This initial increase is due to the fact that during the beginning of the simulation time, the SSs and the BS are busy attempting to complete the ranging process; hence, arriving packets are delayed causing a relatively high average delay. It is also observed that the DFPQ scheduler yields a higher average end-to-end delay for real-time classes than PDFPQ and SS-assisted algorithms. Indeed, DFPQ only focuses on achieving high bandwidth utilization by dynamically dividing the bandwidth between the UL and DL subframes. As for the SS-assisted and PDFPQ schedulers, the average end-to-end delay decreases to a very low and stable value for both rtPS and ertPS classes because of processing the critical packets as soon as they arrive to guarantee the latency QoS requirement of real-time applications. The figure shows also that the SS-assisted scheduler outperforms the PDFPQ scheduler. This can be justified because PDFPQ algorithm assumes that real-time scheduling can occur at the BS for the UL subframe. However, in reality, all packet arrivals occur at the beginning of each frame, which makes preemption not feasible for the UL scheduling at the BS.

4.3.2. Throughput analysis.

In these experiments, we have assumed that a total of N frames are scheduled; then, the throughput, Th_i, with

DOI: 10.1002/wcm



(a) (b)

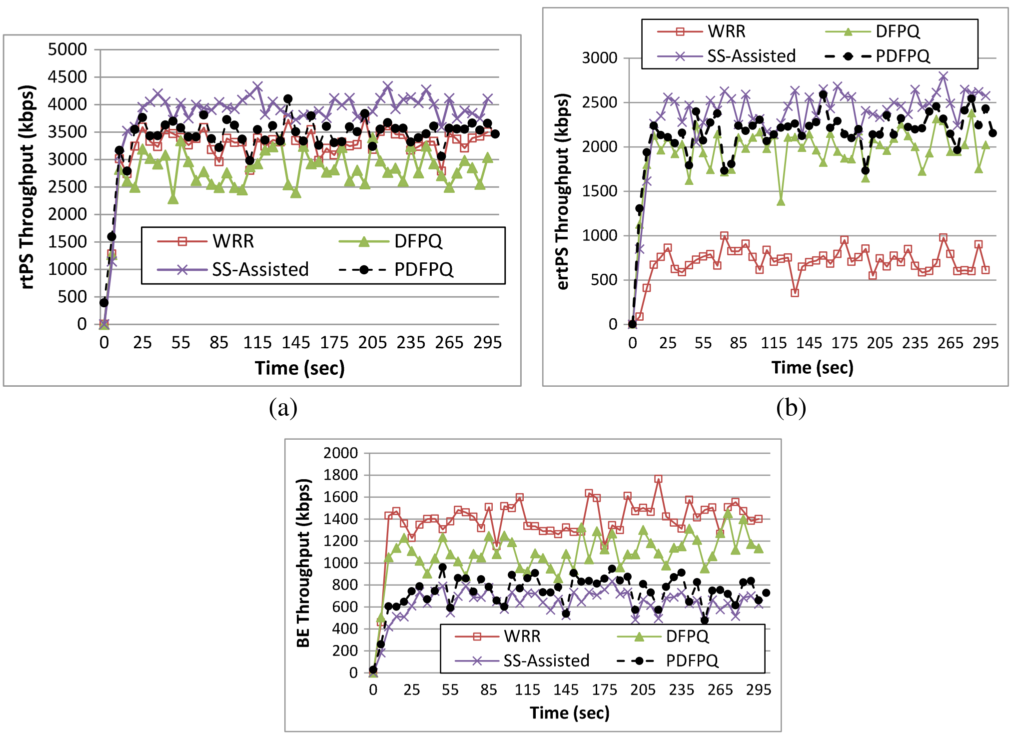
Figure 9. Average end-to-end delay for the three algorithms (a) real-time polling service (rtPS) and (b) extended real-time polling service (ertPS).

respect to time, for each service i was calculated using Equation (12):

$$Th_i = \sum_{j=t_n}^{j=t_{n+1}} packetSize_i \tag{12}$$

where $packetSize_i$ is the size of the serviced data by the service i in frame n between the start time, t_n, of frame n and the start time, t_{n+1}, of the next frame $n + 1$.

The duration of a frame, starting at t_n and ending before t_{n+1}, is 10 ms. Figure 10 compares the results of the four scheduling algorithms for each of the rtPS, ertPS, and BE QoS classes. We can observe that rtPS class throughput is relatively higher than other classes because of its variable packet size and its medium priority. The figure also shows that the WRR-based scheduler services only a small amount of ertPS class, whereas the SS-assisted scheduler



(a) (b)

(c)

Figure 10. Throughput per service class for the three algorithms (a) real-time polling service (rtPS), (b) extended real-time polling service (ertPS), and (c) best effort (BE).

Wirel. Commun. Mob. Comput. 2015; **15**:584–600 © 2013 John Wiley & Sons, Ltd.
DOI: 10.1002/wcm

has the highest throughput for real-time rtPS and ertPS classes. This is normal because the latter processes a larger amount of critical packets and gives them a higher priority to be processed as soon as they arrive to the queues. However, the proposed scheduler has the lowest throughput result for BE class because by processing more real-time critical packets, the remaining bandwidth for the lowest priority classes lessens allowing a less number of BE packets to be processed.

4.3.3. Throughput analysis for critical packets.

For a total of N scheduled frames, the throughput of critical packets, Th_Cr_i, for each service$i$ is computed using Equation (13).

$$Th\_Cr_i = \sum_{j=t_n}^{j=t_{n+1}} criticalPacketSize_i \tag{13}$$

where *criticalPacketSize* is the size of the serviced critical packets by the service i between the start time, t_n, of frame n, and the start time, t_{n+1}, of the next frame $n+1$.

Figure 11 compares the throughput of critical packets for each of the rtPS and ertPS QoS classes in the four scheduling algorithms. It shows that this throughput is relatively stable especially for the WRR and the SS-assisted algorithms. The throughput of rtPS class is relatively higher than that of ertPS because rtPS packets are generated with a variable packet size having a higher priority. Moreover, it shows that WRR, DFPQ, and PDFPQ schedulers service a smaller amount of rtPS and ertPS critical packets than the SS-assisted scheduler, which has the highest throughput because of processing a larger amount of rtPS and ertPS critical packets.

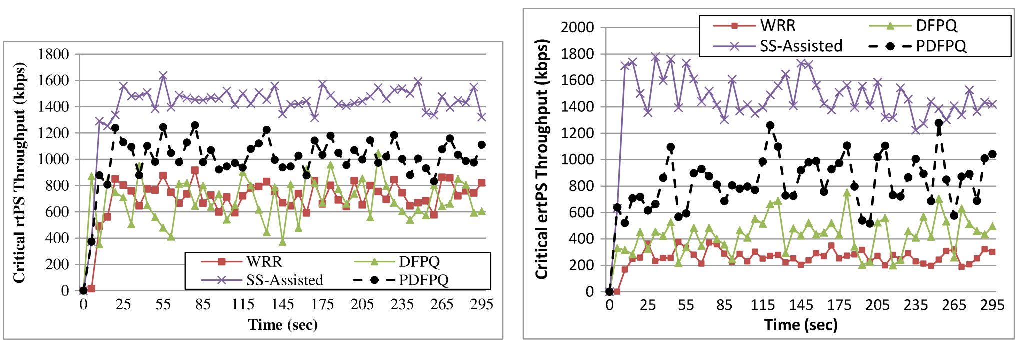
4.3.4. End-to-end delay analysis for critical packets.

The average end-to-end delay, *AverageDelay_Cr*, for critical packets of each real-time service flow i per frame n is computed using Equation (14).

$$AverageDelay\_Cr_i = \left(\frac{\sum_{j=t_n}^{j=t_{n+1}} deliveryTime - \sum_{j=t_n}^{j=t_{n+1}} arrivalTime}{\sum_{j=t_n}^{j=t_{n+1}} receivedCritPacket} \right) \tag{14}$$

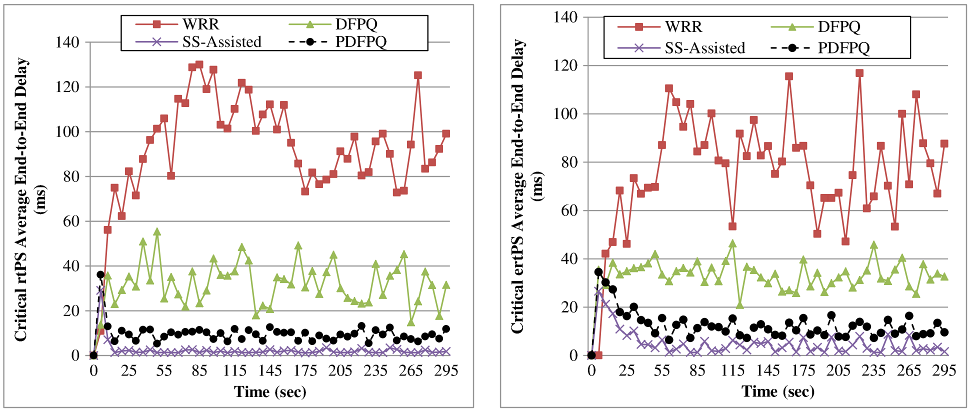
where *arrivalTime* is the time at which a critical packet arrives at the MAC layer of the source node, *deliveryTime* is the time at which the packet is delivered to the MAC layer of the destination node, and *receivedCritPacket* is the total number of critical packets received between the starting time t_n of frame n and the starting time t_{n+1} of the next frame $n+1$.

The WRR scheduler does not control the delay of critical rtPS and ertPS packets, which makes the average delay of these packets higher than the average delay of other serviced packets and exceeds the maximum delay limitation as required by their QoS latency parameters. Thus, we can deduce that these critical packets have a great impact on increasing the average delay of the whole system as shown in Figure 12. As for the DFPQ scheduler, the average end-to-end delay for real-time critical packets, although lower than that of WRR, is still relatively high and close to the ML required for guaranteeing the QoS parameters of the rtPS and ertPS classes. As for the SS-assisted and the PDFPQ schedulers, the delay of critical packets increases at the beginning of the simulation time because of the ranging process then decreases to a very low and stable value,



(a) (b)

Figure 11. Critical packets throughput (a) real-time polling service (rtPS) and (b) extended real-time polling service (ertPS).



(a) (b)

Figure 12. Critical packets average delay (a) real-time polling service (rtPS) and (b) extended real-time polling service (ertPS).

thus guaranteeing the latency QoS requirements of real-time applications. The SS-assisted scheduler outperforms the PDFPQ scheduler because it controls the critical rtPS and ertPS packets at both BS and SS.

4.3.5. Scalability.

The WiMAX architecture consists of a number of SSs associated with a BS through which the SSs communicate. In the real world, the number of SSs that are associated with a BS varies according to the total bandwidth of the BS and the total bandwidth of each SS. For instance, assume that a BS has a total bandwidth of 50 Mbps and each SS has a total bandwidth of 1 Mbps, then a number of 50 SSs utmost can be associated with each BS. In this section, we evaluate the three scheduling algorithms with an increasing number of SS nodes to test their scalability.

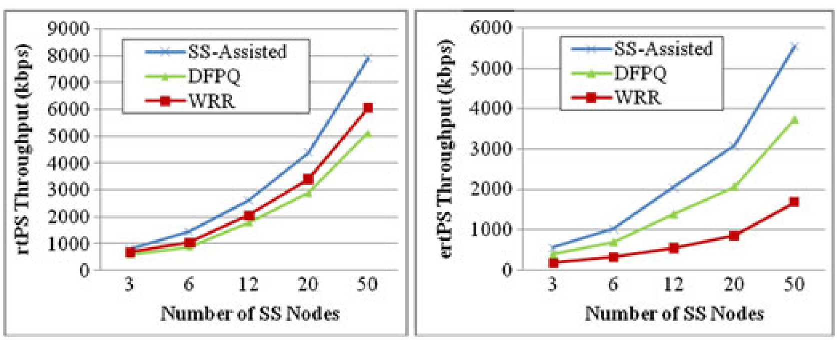
We analyze first the throughput variation of different QoS classes while increasing the number of SS nodes. The total number of bytes delivered throughout the simulation, each time for a different number of SS nodes, is computed using Equation (15).

$$Th = \sum_{j=0}^{j=T} packetSize \qquad (15)$$

where Th is the total throughput of the system from time 0 to time T, the end of simulation.

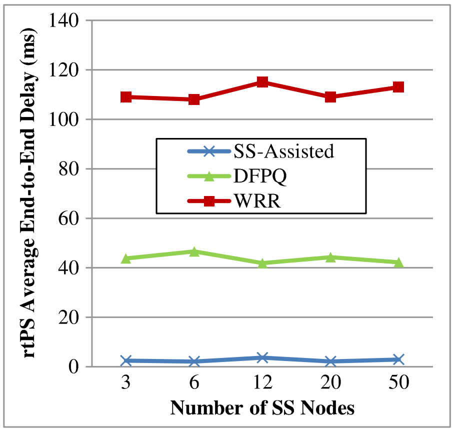
Figure 13 shows that the throughput increases for the different QoS classes in each of the three algorithms as the number of SSs increases. Still, the SS-assisted algorithm has the highest throughput for the real-time classes, rtPS and ertPS.

To evaluate the impact of scalability on delay, we have computed the average end-to-end delay for rtPS and ertPS QoS classes as per Equation (16).

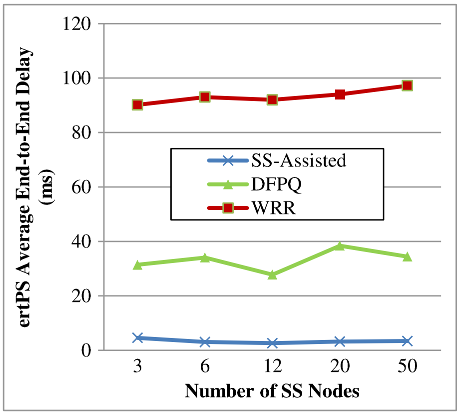


(a) (b)

Figure 13. Throughput in kilobits per second (kbps) versus number subscriber station (SS) nodes (a) real-time polling service (rtPS) and (b) extended real-time polling service (ertPS).



(a)



(b)

Figure 14. End-to-end delay versus number subscriber station (SS) nodes (a) real-time polling service (rtPS) and (b) extended real-time polling service (ertPS).

$$AverageDelay = \left(\frac{\sum_{j=0}^{j=T} deliveryTime - \sum_{j=0}^{j=T} arrivalTime}{\sum_{j=0}^{j=T} receivedPacket} \right) \quad (16)$$

where *arrivalTime* is the time at which a packet arrives at the MAC layer of the source node, *deliveryTime* is the time at which the packet is delivered to the MAC layer of the destination node, and *receivedPacket* is the total number of packets received between the start and the end of the simulation.

Figure 14 shows that both the DFPQ and WRR algorithms do not suppress end-to-end delay for real-time traffic as the number of SS nodes increases. On the other hand, the SS-assisted scheduling algorithm still suppresses the real-time end-to-end delay even with an increasing number of SS nodes and its average end-to-end delay remains bounded by a value of 5 ms for both the rtPS and ertPS classes.

4.3.6. Rate of critical packets.

We have also studied the throughput of the three algorithms while increasing the critical packets rate within the system. Figure 15 shows that the throughput of real-time traffic, namely rtPS and ertPS, increases with the increase of the critical packets rate for all the scheduling algorithms. The figure shows also that WRR processes the least number of ertPS packets while DFPQ processes the least number of rtPS packets. For both rtPS and ertPS, the SS-assisted has the highest throughput. As for the BE QoS class, which has the least priority, WRR performs the best, whereas the SS-assisted algorithm shows a low BE throughput, but it remains stable or slightly decreases with the increase in the rate of critical packets. This is because as the number of critical packets increases, the SS-assisted scheduler ensures that most of these packets are processed by time, which causes a smaller amount of bandwidth to remain for the low priority classes such as the BE to be processed. Still, the decrease in the BE throughput remains stable because the SS-assisted scheduler does not allow low priority classes to starve. This is ensured because the scheduler relies on the *DC* and the Q_{crit} values in its algorithm, which limit amount of real-time and critical packets to be serviced.

4.3.7. Fairness.

One of the major goals of a scheduling algorithm is to maintain fairness among the different queues of the scheduler despite having them prioritized. A scheduling algorithm is said to be fair if it does not allow the lowest priority queue to starve and maintains a bounded difference in normalized services received by different flows in the scheduler. In this section, we study the fairness of the four scheduling algorithms proving that they still maintain a bounded fairness. The formula of fairness, *FAIR*, is defined to compare the throughput between rtPS, which is the highest priority QoS class among the scheduler queues, and BE, which is the lowest QoS class, as follows [6]:

$$FAIR = \left| \frac{Thr_{rtPS}}{Tot_{rtPS}} - \frac{Thr_{BE}}{Tot_{BE}} \right| \quad (17)$$

DOI: 10.1002/wcm

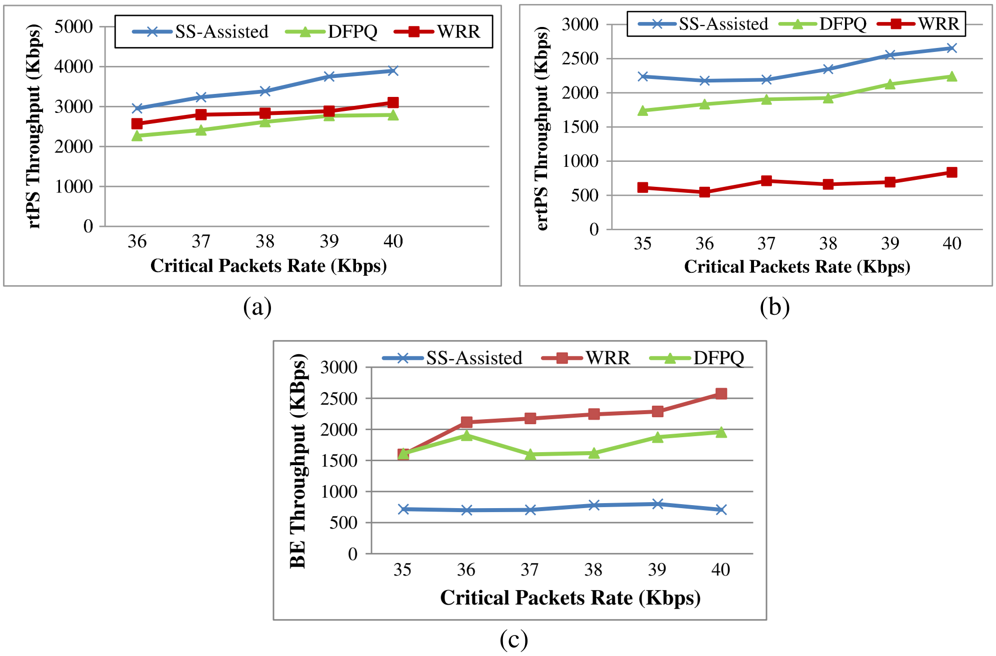


Figure 15. Throughput per service class for the three algorithms versus critical packets rate (a) real-time polling service (rtPS), (b)extended real-time polling service (ertPS), and (c) best effort (BE).

where *Tot_rtPS* is the total rtPS traffic and *Tot_BE* is the total BE traffic generated in the system packets rate within the system. Figure 16 shows that the fairness boundary of the PDFPQ and the SS-assisted scheduling algorithms increase as the data rate of rtPS increases. However, when the data rate reaches a certain point, the scheduling architecture keeps the fairness boundary under 0.15. On the other hand, WRR and DFPQ keep their fairness boundary under 0.04. As a result, it is clear that WRR and DFPQ have a lower fairness boundary than PDFPQ and SS-Assisted scheduling algorithms which is normal since the latter allocate more bandwidth to critical real-time packets and thus a lower bandwidth allocation for non-real packets. Despite this and since the proposed scheduler keeps its fairness bounded by a relatively small value, it still maintains fairness among its different priority queues and does not allow low priority queues, such as the BE class, to starve. As a result, the SS-assisted-based scheduling architecture ensures a better QoS guarantee for real-time applications specifically concerning the ML requirements while still maintaining fairness among the various applications.

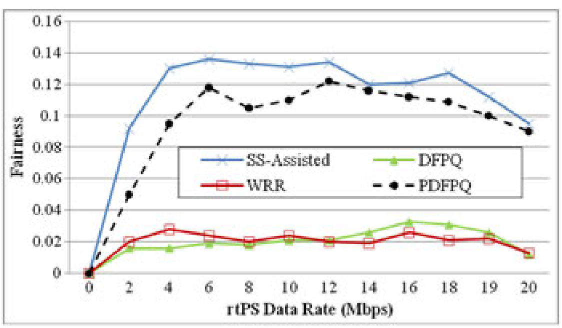


Figure 16. Fairness of the four scheduling algorithms.

5. CONCLUSION

In this paper, we have proposed a preemption-based scheduling algorithm for WiMAX networks that distributes dynamically the bandwidth among service flows while giving real-time packets more chances to satisfy their QoS requirements. The scheduling in the proposed algorithm is assisted by the subscriber stations, hence requiring two schedulers, a scheduler at the BS and another one at the SS, where preemption is applied to both of them. At the SS, preemption is always feasible; hence, a preemptive queue being serviced can be preempted if a packet, which cannot be delivered within its deadline, arrives to a non-preemptive queue. However, at the BS, preemption is only feasible if a critical packet arrives to a non-preemptive DL queue. In this case, the newly arriving packet will be serviced first, and the preempted queue will continue to be serviced afterwards. The proposed algorithm was implemented and integrated with the network simulator NS2 using the MAC layer of the NDSL WiMAX module, then compared its performance with other peer algorithms found in the literature. Simulation scenarios were performed to measure mainly the throughput and the average end-to-end delay of the different QoS classes, specifically real-time classes, for four different scheduling algorithms. The proposed approach has shown better performance than the other algorithms in terms of enhancing the throughput and the average delay of real-time QoS classes.

ACKNOWLEDGEMENTS

This work was supported in part by a grant from the Lebanese National Council For Scientific Research (NO. 111135 522247, LNCSR/2010).

REFERENCES

1. IEEE Std 802.16e-2006. IEEE Standard for Local and Metropolitan Area Networks Part 16: Air Interface for Fixed Broadband Wireless Access Systems, February 2006.
2. IEEE Std 802.16-2009. IEEE standard for local and metropolitan area networks part 16: Air interface for fixed and mobile broadband wireless access systems. *Revision of IEEE Std 802.16-2004 Technical report*, 2009.
3. Safa H, Khayat S. A distributed scheduling algorithm for mobile WiMAX networks, In *Proceedings of the IEEE International Conference on Selected Topics in Mobile and Wireless Networking (ICOST'2011)*, Shanghai, China, October 2011; 94–99.
4. Chen J, Wang C, Tsai D, Chang C, Liu S, Guo J, Lien W, Sum J, Hung C. The design and implementation of WiMAX module for ns-2 simulator, In *Workshop on ns-2 (WNS2): The IP network simulator*, Piza, Italy, October 2006; 5.
5. Abu Ali N, Dhrona P, Hassanein H. A performance study of uplink scheduling algorithms in point-to multipoint WiMAX networks. *Computer Communications* 2009; **32**(3): 511–521.
6. Chen J, Jiao W, Wang H. A Service flow management strategy for IEEE 802.16 broadband wireless access systems in TDD Mode, In *Proceedings of the IEEE International Conference on Communications, (ICC 2005)*, Seoul, Korea, May 2005; 3422–3426.
7. Chu G, Wang D, Mei S. A QoS architecture for the MAC protocol of the IEEE 802.16BWA system, In *Proceedings of the IEEE Conference on Communications, Circuits, and Systems*, Chengdu, China, June 2002; 435–439.
8. Hawa M, Petr DW. Quality of service scheduling in cable and broadband wireless access systems, In *Proceedings of the 10th IEEE International Workshop on Quality of Service*, Miami Beach, USA, May 2002; pp. 247–255.
9. Safa H, Artail H, Karam M, Soudah R, Khayat S. A new scheduling architecture for IEEE 802.16 wireless metropolitan area network, In *Proceedings of The fifth ACS/IEEE International Conference on Computer Systems and Applications, AICCSA '2007*, Amman, Jordan, May 2007; 203–210.
10. Sayenko A, Alanen O, Karhula J, Hamalainen T. Scheduling solution for the IEEE 802.16 base station. *Computer Networks* 2008; **52**(1): 96–115.
11. Shreedhar M, Varghese G. Efficient fair queueing using deficit round Robin, In *Proceedings of the ACM SIGCOMM'95*, Cambridge, Massachusetts, USA, August 1995; pp. 231–242.
12. So-In C, Jain R, Tamimi A. A deficit round robin with fragmentation scheduler for IEEE 802.16e mobile WiMAX, In *Proceedings of the 32nd International Conference on Sarnoff Symposium*, Princeton, New Jersey, USA, March 2009; pp. 1–7.
13. Tokel TB, Aktas D. A low-complexity transmission and scheduling scheme for WiMAX systems with base station cooperation. *EURASIP Journal on Wireless Communications and Networking* 2010; **2010**(7): 1–10.
14. Hahne EL, Gallager RG. Round Robin scheduling for fair flow control in data communication networks, In *Proceedings of the IEEE International Conference on Communications*, Toronto, Canada, June 1986.
15. Cicconetti C, Erta A, Lenzini L, Mingozzi E. Performance evaluation of the IEEE 802.16 MAC for QoS support. *IEEE Transactions on Mobile Computing* 2007; **6**(1): 26–38.
16. Georgiadis L, Guerin R, Parekh A. Optimal multiplexing on a single link: delay and buffer requirements. *IEEE Transactions on Information Theory* 1997; **43**(5): 1518–1535.
17. Demers A, Keshav S, Shenker S. Analysis and simulation of a fair queueing algorithm. *ACM SIGCOMM Computer Communication Review* 1989; **19**(4): 1–12.
18. Moraes L, Rubin I. Message delays for a TDMA scheme under a nonpreemptive priority discipline. *IEEE Transaction on Communications* 1984; **32**(5): 583–588.
19. The Network Simulator 2 (NS-2). (Available from: http://nsnam.isi.edu/nsnam/index.php/User_Information) [accessed on 5 November 2011].
20. Pitic R, Serrelli F, Redana S, Capone A. Performance evaluation of utility-based scheduling schemes with QoS guarantees in IEEE 802.16/WiMAX systems. *Wireless Communications and Mobile Computing* 2010; **10**(7): 912–931.
21. Andel TR, Yasinsac A. On the credibility of MANET simulations. *IEEE Computer* 2006; **39**(7): 48–54.

AUTHORS' BIOGRAPHIES

Haidar Safa received a B.Sc. in Computer Science in 1991 from Lebanese university, Lebanon, M.Sc. in Computer Science in 1996 from University of Quebec at Montreal (UQAM), and a Ph.D. in Electrical and Computer Engineering in 2001 from Ecole Polytechnique de Montreal. He joined ADC Telecommunications and SS8 Networks in 2001 where he worked on designing and developing networking and system software. In 2003, he joined the American University of Beirut where he is currently an associate professor at the Department of Computer Science. Dr. Safa is also associated with the Mobile Computing and Networking Research Laboratory (LARIM), Ecole Polytechnique de Montreal, Montreal, Canada. His research interests include mobile and wireless networks, distributed computing, quality of service, routing, and network security.

Samar Khayat received a B.Sc. in Computer Science in 2005 from the American University of Beirut, and M.Sc. in Computer Science in 2009 from the American University of Beirut in Lebanon. She joined TerraNet in 2005, as a senior software designer and developer. Her research interests include mobile and wireless networks, software engineering, and mobile application development.

Wirel. Commun. Mob. Comput. 2015; **15**:584–600 © 2013 John Wiley & Sons, Ltd.
DOI: 10.1002/wcm