INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

Uplink Scheduler for WiMAX Networks

J. F. Borin *N. L. S. da Fonseca*

Technical Report - IC-11-22 - Relatório Técnico

December - 2011 - Dezembro

Uplink Scheduler for WiMAX Networks

Juliana Freitag Borin   Nelson Luis Saldanha da Fonseca*

Abstract

In order to support real time and high-bandwidth applications the IEEE 802.16 standard is expected to provide Quality of Service (QoS). Although the standard defines a QoS signaling framework and five service levels, scheduling disciplines are unspecified. In this technical report, we introduce a scheduling scheme for the uplink traffic. The proposed solution is fully standard compliant and can be easily implemented in the base station. Simulation results show that this solution is able to meet the QoS requirements of multimedia applications.

1 Introduction

To support the diversity of multimedia applications available on the Internet, the IEEE 802.16 standard [1] defines four types of service flows and mechanisms for bandwidth requests by connections in the uplink direction. The first service flow, Unsolicited Grant Service (UGS), periodically receives fixed size grants without the need to request them. The second, real-time Polling Service (rtPS), provides periodic unicast bandwidth request opportunities to subscribers; these opportunities ensure latency bound and minimum traffic rate guarantees. The non-real-time Polling Service (nrtPS) offers periodic unicast bandwidth request opportunities, but using more spaced intervals than rtPS, as well as minimum traffic rate guarantee. The fourth type of service flow, Best Effort (BE), shares contention bandwidth request opportunities with the nrtPS service flow. The IEEE 802.16e-2005 [2], an amendment to [1], adds a new service flow called extended real time Polling Service (ertPS). This service uses a grant mechanism similar to the one used to support UGS connections, except that periodically allocated grants can be used to send bandwidth requests to inform the required grant size.

---

*Institute of Computing, University of Campinas, Campinas, SP, Brazil, 13089-971. This research was partially sponsored by UOL, through its UOL Bolsa Pesquisa program, process number 20071211114200, and by CNPq.

Part of this paper was published in J. F. Borin, and N. L. S. Fonseca, *Uplink Scheduler and Admission Control for the IEEE 802.16 Standard*. In Proceedings of the IEEE Global Telecommunications Conference - GLOBECOM, pp. 1-6, 2009. DOI `http://dx.doi.org/10.1109/GLOCOM.2009.5425779`

Although these five types of service flows furnish the basis for QoS provisioning, the core of the task resides in resource allocation done by the scheduling mechanism. The IEEE 802.16 standard states that both the downlink and the uplink schedulers at the base station (BS) as well as the scheduler at the subscriber station (SS) should be defined by proprietary implementations. The major challenges, however, are related to scheduling in the uplink direction at the BS. Scheduling at the downlink direction and at the SSs are less complex since information about the queues status is locally available. To develop a standard-compliant uplink scheduler, the following aspects should be taken into account:

1. based on the QoS parameters of the connections and on the bandwidth requests of each SS, the BS should distribute the uplink bandwidth such that QoS requirements of each connection are satisfied.

2. the BS should allocate bandwidth not only for data transmission but also for bandwidth requests transmission according to the request mechanism defined for each type of service flow.

3. the uplink scheduler should support all QoS parameters defined by the standard.

This technical report introduces a standard-compliant mechanism for uplink scheduling to deal with the above mentioned aspects. The scheduler allocates grants with size up to the *maximum traffic burst* value so that connections transmit at rates varying between the *minimum reserved traffic rate* and the *maximum sustained traffic rate*. Moreover, the *maximum latency* requirements of real time connections are guaranteed to the traffic flows that do not exceed the *minimum reserved traffic rates*, as specified by the standard. UGS and ertPS connections receive grants in intervals defined by the *unsolicited grant interval* parameter. Periodic grants for bandwidth requests by rtPS and nrtPS connections are allocated according to the *unsolicited polling interval* parameter.

The proposed scheduler extends previous work by the authors. The versions presented in [9] and in [10] do not include the *maximum sustained traffic rate* and the *maximum traffic burst* policing. Moreover, the scheduler tries to serve all rtPS packets before the expiration of their deadlines, which demands complex admission control schemes in order to guarantee a *maximum latency* value while achieving high levels of utilization. In the scheduler introduced in this technical report, the policing of both parameters is performed by a *dual leaky bucket* regulator. This technique simplifies the algorithm and its configuration, given that it is not necessary to investigate the window size that leads to the best performance for the different types of traffic. Moreover, the scheduler presented here guarantee *maximum latency* to the flows that do not exceed their *minimum reserved traffic rates*. In this way, standard compliance is achieved.

Simulation results show that the proposed mechanism is capable to provide the connection-level QoS support defined by the IEEE 802.16 standard. The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 present the scheduling mechanism. Section 4 describes the simulation environment used to test the mechanisms. Section **??** presents numerical results. Finally, Section 5 concludes the technical report.

2 Related work

Many solutions proposed for the IEEE 802.16 uplink scheduler [5, 8, 14] combine classic scheduling policies developed for wired networks, such as Strict Priority, Weighted Fair Queuing (WFQ), and Earliest Deadline First (EDF), resulting in complex scheduling schemes. More recent work [4, 19] as well as the scheduler proposed in this paper have invested in simple implementable ideas. Given that the uplink scheduler at the BS is executed at each frame and there can be as many as 400 frames per second, simpler solutions become more attractive. Different from the schedulers proposed by Sayenko *et al* [4] and Bai *et al* [19], the scheduler introduced here is fully standard-compliant. The scheduler proposed in [4] does not provide maximum latency guarantees, while the one proposed in [19] uses a priority value computed by the SSs to provide rate and latency guarantees. Although this approach leads to a less complex scheduler in the BS, it restricts the interoperability between equipments from different vendors since all the SSs should be able to calculate the priority values.

3 Scheduling mechanism

The proposed uplink scheduler uses three queues: low, intermediate and high priority queues. The scheduler serves the queues according to their level of priority. The low priority queue stores the BE bandwidth requests. The intermediate queue stores the bandwidth requests sent by both rtPS and nrtPS connections. These requests can migrate to the high priority queue to guarantee that their QoS requirements are met. In addition to the requests migrating from the intermediate queue, the high priority queue stores periodic grants and unicast request opportunities that must be scheduled in the following frame. The BS executes the uplink scheduler at every frame, and it broadcasts the scheduling agenda to the SSs in the UL-MAP message.

At each frame, the scheduler generates periodic grants and inserts them into the high priority queue at predefined intervals. In this way, UGS and ertPS grants are guaranteed, and rtPS and nrtPS unicast request opportunities are provided as specified by the standard. To guarantee the *maximum latency* requirement, the BS assigns a deadline for each rtPS bandwidth request in the intermediate queue. Each time the scheduler is executed, requests with deadlines expiring two frames ahead and associated with connections which have not received the *minimum reserved traffic rate* in a window with duration T migrate from the intermediate queue to the high priority queue. In this way, the scheduler guarantees the maximum latency for the flows that do not exceed the minimum reserved traffic rate, as specified in the standard. It is necessary to know the arrival time of the packets at SS queues to determine the deadline of the requests. Since the BS has no access to this information, it considers that the packet arrived at the queue immediately after the last bandwidth request sent by the connection. Hence, the deadline of a request is equal to the arrival time of the last request sent by the corresponding connection plus the maximum latency requirement of that connection.

The scheduler guarantees the *minimum reserved traffic rate* to the rtPS and to the nrtPS services in windows of duration T. Every time the scheduler is executed, it computes a priority value for each request at the intermediate queue, considering the per connection: *minimum reserved traffic rate*, backlogged requests, and traffic rate received in the current window. Low priority values are assigned to the requests of connections which have already received the *minimum reserved traffic rate* in the current window. For the remaining requests, the lower the rate received by the connection, the higher is the priority value associated to them.

Since all the allocations for a single SS are placed in one grant in the UL-MAP, the scheduler guarantees that the sum of the bandwidth allocated to a single connection is less than or equal to the *maximum traffic burst* requirement. Moreover, the scheduler does not allocate bandwidth for a connection if it results in violation of the *maximum sustained traffic rate*. A dual leaky bucket (Figure 1) is used for maximum burst and maximum rate policing. The capacity of the first bucket ($bucket1$) is infinite and its leaky rate is equal to the *maximum sustained traffic rate*. The second bucket ($bucket2$) has leaky bucket equal to the *maximum traffic burst*. Therefore, before allocating a grant, the scheduler checks whether the grant size is less or equal to the $bucket2$ size.

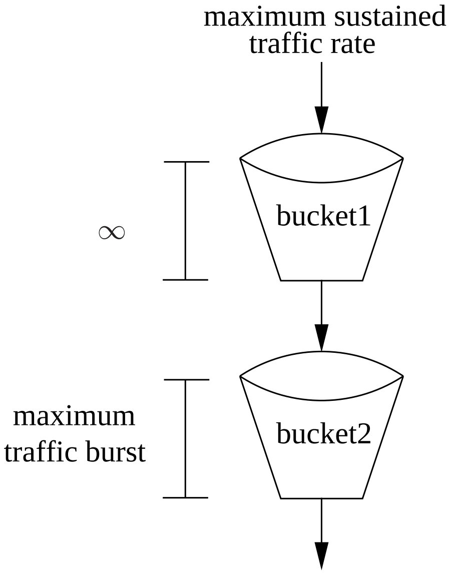

Figure 1: Dual leaky bucket

The Algorithm *Scheduling* presents the proposed scheme. After inserting periodic grants in the high priority queue, the algorithm checks which rtPS and nrtPS requests should migrate from the intermediate queue to the high priority queue (lines 2 and 3). In line 4, the scheduler distributes the non-allocated bandwidth among the BE connections. At the final step, it serves all the requests at the high priority queue.

In the *CheckDeadline* procedure, the scheduler tries to migrate the rtPS requests from the intermediate queue to the high priority queue using the $MigrateBWRequest$ procedure whenever the following conditions happen: i) there is available bandwidth ii) the deadline

Algorithm *Scheduling*

```
insert, in the high priority queue, the periodic data grants and unicast request opportunities that must be scheduled in the next frame;
CheckDeadline;
CheckMinimumBandwidth;
DistributeFreeResources;
schedule the requests in the high priority queue starting from the head of the queue;
```

Procedure *CheckDeadline*

```
for each request $i$ at the intermediate queue do
  if availableBw = 0 then
    break
  if service[CID] = rtPS then
    frame[i] = $\lfloor$(deadline[i] - currentTime)$\div$ frameDuration$\rfloor$;
    if frame[i] = 3 and TwndTR[CID] $<$ minTR[CID]
      MigrateBWRequest(i);
```

Procedure *CheckMinimumBandwidth*

```
for each connection of type rtPS or nrtPS do
  backlogged_tmp[CID] = backlogged[CID]
  TwndTR_tmp[CID] = TwndTR[CID]
  bucket2_tmp[CID] = bucket2[CID]
for each request $i$ at the intermediate queue do
  if minTR[CID] $\leq$ TwndTR_tmp[CID] or bucket2_tmp[CID] = 0 then
    priority[i] = 0;
  else
    priority[i] = backlogged_tmp[CID] $-$ ( TwndTR_tmp[CID] $-$ minTR[CID]);
    TwndTR_tmp[CID] = TwndTR_tmp[CID] + BR[i]
    bucket2_tmp[CID] = bucket2_tmp[CID] + BR[i]
    backlogged_tmp[CID] = backlogged_tmp[CID] $-$ BR[i]
sort the intermediate queue
For each request $i$ at the intermediate queue do
  if availableBw = 0 then
    break
  MigrateBWRequest(i);
```

Procedure *DistributeFreeResources*

```
for each request $i$ at the low priority queue do
  if availableBw = 0 then
    break
    MigrateBWRequest(i);
```

```
PROCEDURE MigrateBWRequest(i)
1.  if BR[i] $>$ availableBw then
2.    grantSize = availableBw;
3.  else
4.    grantSize = BR[i];
5.  if grantSize $>$ bucket2[CID]
6.    grantSize = bucket2[CID];
7.  if $0 <$ grantSize $<$ BR[i] then
8.    create a new request j for connection CID with BR[j] = BR[i] $-$ grantSize;
9.    insert request j in the end of the intermediate queue;
10. BR[i] = grantSize;
11. move request $i$ to high priority queue;
12. TwndTR[CID] = TwndTR[CID] + grantSize;
13. bucket2[CID] = bucket2[CID] $-$ grantSize;
14. backlogged[CID] = backlogged[CID] $-$ grantSize;
15. availableBw = availableBw $-$ grantSize;
```

of a rtPS request expires during the frame following the next one, and iii) the corresponding connection has not received the minimum reserved traffic rate ($minTR[CID]$) in the current window.

The *MigrateBWRequest* procedure checks whether or not the amount of bandwidth solicited by the migrating request ($BR[i]$) is available in the uplink subframe. Whenever the available bandwidth is less than the requested one, it will be allocated and the allocation of the remaining amount solicited will be deferred to a future time. The allocation of part of the requested bandwidth will also be deferred in case it results in violation of the *maximum traffic burst* value. Once the amount of bandwidth that can be allocated to the request is defined (*grantSize*), if this value is less than the $BR[i]$ value, a new request j is created with $BR[j]$ equals to $BR[i] - grantSize$ and it is inserted at the end of the intermediate queue. The bandwidth solicited by the i^{th} request is updated to $grantSize$ and the request is migrated to the high priority queue. Finally, the procedure updates: i) the traffic rate received by the corresponding connection during a window of size T ($TwndTR[CID]$), ii) the size of the $bucket2$ of the corresponding connection ($bucket2[CID]$), iii) the number of bytes requested by the backlogged requests sent by the corresponding connection ($backlogged[CID]$), and iv) the number of available bytes in the uplink subframe ($availableBW$).

The *CheckMinimumBandwidth* procedure first calculates a priority value for each request in the intermediate queue (lines 5-9). Then, it sorts the intermediate queue according to the priority values (line 3). Finally, the scheduler tries to migrate requests to the high priority queue using the *MigrateBWRequest* procedure.

The $DistributeFreeResources$ procedure distributes the available bandwidth among the BE requests by migrating some of them from the low priority queue to the high priority queue.

4 Simulation Experiments

To conduct this study, an ns-2 module for IEEE 802.16 networks was developed [11] [12].

The simulated network consists of a BS, with the SSs uniformly distributed around it. The frame duration is set to 5 ms and the capacity of the channel is assumed to be 40 Mbps with a 1:1 downlink-to-uplink TDD split.

To eliminate the impact of packet scheduling at the SSs on uplink scheduling, each SS has only one service flow. Moreover, an admission control mechanism is used [13], so that results are not affected by excessive number of connections in the network. We consider five different types of traffic: voice, voice with silence suppression, video, FTP and WEB, which are associated with UGS, ertPS, rtPS, nrtPS, and BE services, respectively.

The voice model used was an exponential "on/off" model with mean duration of the "on" and of the "off" periods equals to 1.2 s and 1.8 s, respectively. During the "on" periods, 66-byte packets are generated at every 20-ms [15]. The voice with silence suppression model used the Enhanced Variable Rate Codec (EVRC) [3], with packets generated every 20 ms employing Rate 1 (171 bits/packet), Rate 1/2 (80 bits/packet), Rate 1/4 (40 bits/packet) or Rate 1/8 (16 bits/packet). Video traffic was generated by real MPEG traces [17]. The WEB traffic was modeled by a hybrid Lognormal/Pareto distribution, with the body of the distribution corresponding to an area of 0.88 modeled by a Lognormal distribution with a mean of 7247 bytes and the tail modeled by a Pareto distribution with mean of 10558 bytes [16]. FTP traffic was generated using an exponential distribution with a mean of 512 KBytes.

The unsolicited grant interval for the UGS and for the ertPS service is 20 ms. The unsolicited polling interval for the rtPS service is 20 ms and for the nrtPS service is 1 s.

The maximum latency requirement of the rtPS service is 100 ms and each connection has its own minimum reserved traffic rate and maximum sustained traffic rate requirements (which vary according to the mean rate of the transmitted video). The nrtPS service has minimum reserved traffic rate requirement of 200 Kbps, and maximum sustained traffic rate requirement of 300 Kbps. The BE service does not have any QoS requirement.

The lifetimes of connections are exponentially distributed with average equal to 600 s for the rtPS connections and equal to 300 s for the UGS, ertPS, and nrtPS connections. The connection arrival rates are governed by an exponential distribution with the mean varying from 60 s to 2 s for each type of service. In this way, under the lowest load, on average, one connection of each type of service arrives per minute, while under the highest load, an average of 30 connections of each type of service arrive per minute.

The number of BE connections is equal to 20 in all the simulated situations. Given that the duration of each simulation was 3600 s and that the BE connections are always admitted, the constant admission of new BE connections would result in an excessive number of connections in the network.

Each result was produced by running the simulation ten times with different seeds. The mean values and the 95% confidence intervals are shown in the figures.

The following results compare the service parameters measured in the simulations with the corresponding QoS requirements defined for each service flow. The aim is to verify the capability of the proposed admission control and scheduler to support QoS at the connection level. Specially, it is expected that: 1) UGS and ertPS connections receive periodic data grants according to the *unsolicited grant interval*; 2) traffic not exceeding the *minimum reserved traffic rate* of an rtPS connection is delayed less than the *maximum latency* constraint of the connection; and 3) the average throughput of any rtPS or nrtPS connection is no less than its *minimum reserved traffic rate* and no more than its *maximum sustained traffic rate*.

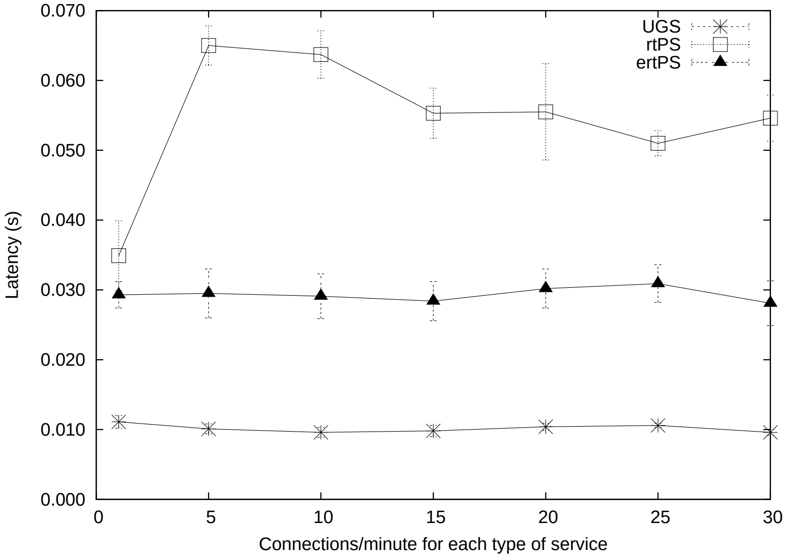

Figure 2: Average latency of UGS, ertPS and rtPS services

Figure 2 shows the average latency for the UGS, the ertPS, and the rtPS connections. The latency values of the UGS and of the ertPS connections were not affected by the increase of the connection arrival rates, which shows that the uplink scheduler is able to provide data grants at fixed intervals as required by the standard. The latency of the rtPS connections did not surpass the required value of 100 ms. The highest latency values are observed when the connection arrival rate is equal to 5 connections of each type per minute, that is the load under which there is the highest number of admitted rtPS connections. When the connection arrival rate is higher than 5 connections of each type of service per minute, the rtPS connections are more likely to be rejected than the connections associated with the other type of services, as previously explained. The utilization of the intermediate queue is high when the number of accepted rtPS connections is also high, since the rtPS connections send bandwidth requests at every 20 ms while the nrtPS connections send bandwidth requests every 1 s, which yields to high latency values.

Figures 3, 4, 5 show the average throughput of the rtPS connections with minimum reserved traffic rate of 200 Kbps, 500 Kbps, and 900 Kbps, respectively, and Figure 6 shows the average throughput of the nrtPS connections. In all the simulated scenarios, the average throughput values of the rtPS and of the nrtPS connections were in the range defined by the minimum and the maximum rates requirements as specified by the standard.

The throughput of the nrtPS connections, measured at the MAC layer, was a little higher

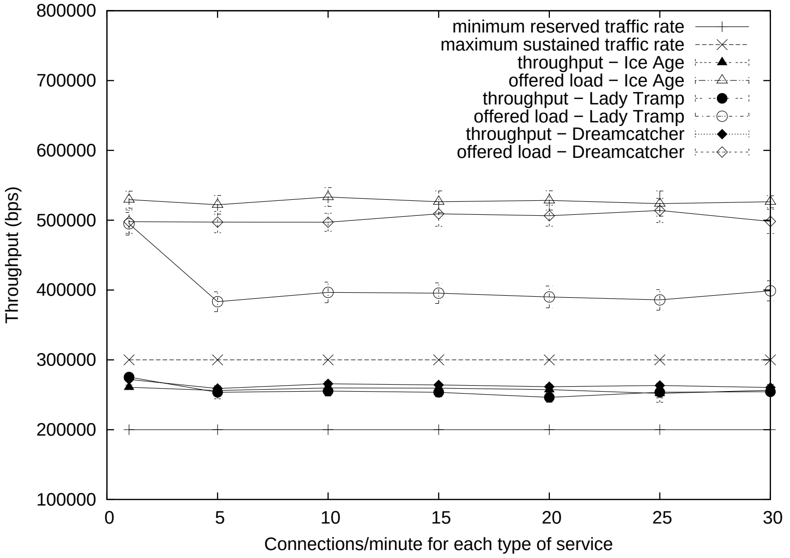

Figure 3: Average throughput of rtPS connections with minimum rate requirement of 200 Kbps

than the offered load, measured at the transport layer, due to additional load incurred by MAC headers. Although the nrtPS flows were configured to generate an average rate equals to the maximum sustained traffic rate, the joint effect of the maximum rate control done by the scheduler and the TCP congestion control resulted in an offered load lower than 300 Kbps. Consequently, all the nrtPS traffic generated by the upper layers was served at the MAC layer in all the simulated scenarios. The interaction between the scheduling mechanism and the TCP congestion control mechanism shall be investigated in the future to counteract the resource underutilization side effect.

Although not shown here due to space limitations, in all the simulated scenarios the BE connections were able to transmit in the slots not used by the higher priority service flows.

5 Conclusions

This technical report introduced a new standard-compliant scheduler for the uplink traffic in IEEE 802.16 networks. The proposed scheduler provides maximum latency and minimum rate guarantees without violating the maximum sustained traffic rate and the maximum traffic burst values. Simulation results show that the proposed solution is able to provide QoS for the different types of service defined by the IEEE 802.16 standard, yet being standard-compliant.

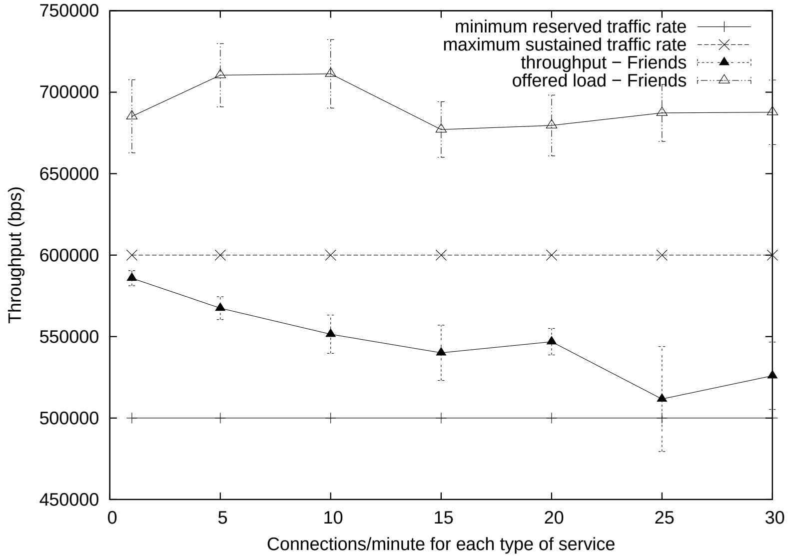

Figure 4: Average throughput of rtPS connections with minimum rate requirement of 500 Kbps

References

[1] IEEE Standard 802.16-2004, Part 16. *Air Interface for Fixed Broadband Wireless Access Systems*, 2004.

[2] IEEE Standard 802.16-2005, Part 16. *Air Interface for Fixed and Mobile Broadband Wireless Access Systems*, 2005.

[3] 3GPP2 C.S0014-0, *Enhanced Variable Rate Codec (EVRC).*

[4] A. Sayenko, O. Alanen, and T. Hamalainen, *Scheduling solution for the IEEE 802.16 base station.* Computer Networks, vol. 52, no. 1, pp. 96-115, 2008.

[5] D. Tarchi, R. Fantacci, and M. Bardazzi, *Quality or Service management in IEEE 802.16 wireless metropolitan area networks.* In Proceedings of the IEEE International Conference on Communications, pg. 1789-1794.

[6] D. Niyato, and E. Hossain, *QoS-aware bandwidth allocation and admission control in IEEE 802.16 broadband wireless access networks: A non-cooperative game theoretic approach.* Computer Networks, vol. 55, no. 11, pg. 3305-3321.

[7] H. Wang, B. He, and D. P. Agrawal, *Above packet layer level admission control and bandwidth allocation for IEEE 802.16 wireless MAN.* Simulation Modeling Practice and Theory, vol. 15, no. 14, pg. 266-382.

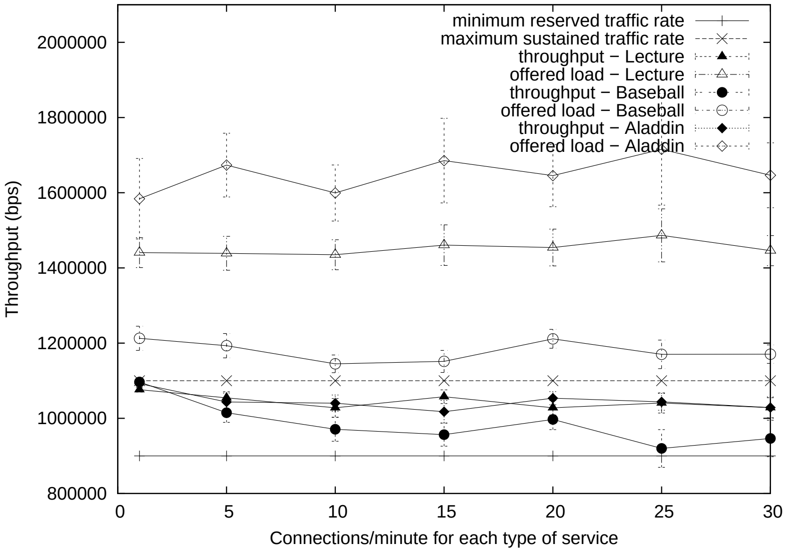

Figure 5: Average throughput of rtPS connections with minimum rate requirement of 900 Kbps

[8] J. Chen, W. Jiao, and H. Wang, *A Service Flow Management Strategy for IEEE 802.16 Broadband Wireless Access Systems in TDD Mode.* In Proceedings of the IEEE ICC, pp. 3422-3426, 2005.

[9] J. Freitag, and N. L. S. Fonseca, *Uplink Scheduling with Quality of Service in IEEE 802.16 Networks.* In Proceedings of the IEEE Global Telecommunications Conference - GLOBECOM, pp. 2503-2508, 2007. DOI `http://dx.doi.org/10.1109/GLOCOM.2007.476`

[10] J. F. Borin, and N. L. S. Fonseca, *Scheduler for IEEE 802.16 networks.* IEEE Communications Letters, vol. 12, no. 4, pp. 274-276, 2008. DOI `http://dx.doi.org/10.1109/LCOMM.2008.072110`

[11] J. Freitag, and N. L. S. Fonseca, *WiMAX Module for the ns-2 Simulator.* In Proceedings of the IEEE 18th International Symposium on Personal, Indoor and Mobile Radio Communications - PIMRC, pp. 1-6, 2007. DOI `http://dx.doi.org/10.1109/PIMRC.2007.4394155`

[12] J.F. Borin, and N.L.S. Fonseca, *Simulator for WiMAX networks.* Simulation Modeling Practice and Theory, vol. 16, no. 7, pp. 817-833, 2008. DOI `http://dx.doi.org/10.1016/j.simpat.2008.05.002`

[13] J. F. Borin, and N. L. S. Fonseca, *Uplink Scheduler and Admission Control for the IEEE 802.16 Standard.* In Proceedings of the IEEE Global Telecommunications Conference - GLOBECOM, pp. 1-6, 2009. DOI `http://dx.doi.org/10.1109/GLOCOM.2009.5425779`

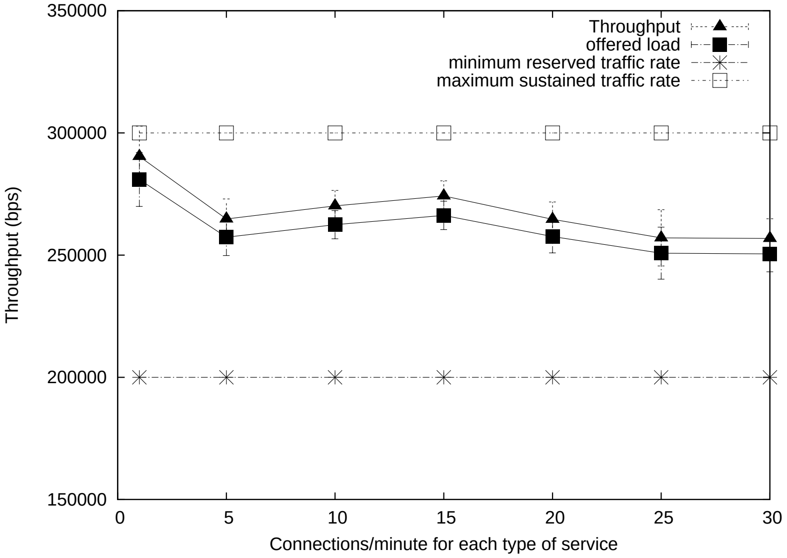

Figure 6: Average throughput of nrtPS connections

[14] K. Wongthavarawat and A. Ganz, *IEEE 802.16 based last mile broadband wireless military networks with Quality of Service support.* In Proceedings of the IEEE MILCOM'03, pp. 779-784, 2003.

[15] P. Brady, *A model for generating On-Off speech patterns in Two-Way conversations.* Bell System Tech. Journal, vol. 48, pp. 2445-2472, 1969.

[16] P. Barford et all, *Changes in web client access patterns: Characteristics and caching implications.* Tech. Report 1998-023, Boston Uni., 1998

[17] P. Seeling, M. Reisslein, and B. Kulapala, *Network Performance Evaluation Using Frame Size and Quality Traces of Single-Layer and Two-Layer Video: A Tutorial.* IEEE Communications Surveys and Tutorials, vol. 6, no. 2, pp. 58-78, 2004.

[18] S. Chandra, and A. Sahoo, *An efficient call admission control for IEEE 802.16 networks.* In Proceedings of the 15th IEEE Workshop on Local & Metropolitan Area Networks, pg. 188-193.

[19] X. Bai, A. Shami, and Y. Ye, *Robust QoS control for single carrier PMP mode IEEE 802.16 systems.* IEEE Transactions on Mobile Computing, vol. 7, no. 4, pg. 416-429.