Publication P5

Jani Lakkakorpi, Alexander Sayenko, and Jani Moilanen. 2008. Comparison of different scheduling algorithms for WiMAX base station: Deficit Round-Robin vs. Proportional Fair vs. Weighted Deficit Round-Robin. In: Proceedings of the 2008 IEEE Wireless Communications and Networking Conference (WCNC 2008). Las Vegas, Nevada, USA. 31 March - 3 April 2008, pages 1991-1996.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the WCNC 2008 proceedings.

Comparison of Different Scheduling Algorithms for WiMAX Base Station

Deficit Round-Robin vs. Proportional Fair vs. Weighted Deficit Round-Robin

Jani Lakkakorpi
Nokia Technology Platforms
Helsinki, Finland
jani.lakkakorpi@nokia.com

Alexander Sayenko
Nokia Research Center
Helsinki, Finland
alexander.sayenko@nokia.com

Jani Moilanen
Nokia Siemens Networks
Espoo, Finland
jani.moilanen@nsn.com

***Abstract*—In this paper, we present a performance comparison of different WiMAX base station (BS) scheduling algorithms: Deficit Round-Robin (DRR) vs. Proportional Fair (PF) vs. Weighted Deficit Round-Robin (WDRR). Our simulations show that when the radio channel conditions are taken into account (in PF and WDRR schedulers), the improvements in throughput can be considerable.**

***Key Words*—IEEE 802.16 WiMAX, scheduling, QoS, ns-2**

I. INTRODUCTION

WiMAX is an IEEE standard (IEEE 802.16d/e) for wireless broadband access network [1, 2]. Some of the main advantages of WiMAX over other access network technologies are longer range and more sophisticated support for Quality of Service (QoS) at the MAC level. Several different types of applications and services can be used in WiMAX networks and the MAC layer is designed to support this convergence. The standard defines two basic operational modes: mesh and point-to-multipoint (PMP). In the former mode, subscriber stations (SS) can communicate to each other and to the base stations (BS). In the PMP mode, the SSs are only allowed to communicate through the BS. Thus, the provider can control the environment to ensure the QoS requirements of its customers.

There can be multiple separate connections between the BS and an SS. At the BS, all downlink (DL) connections have dedicated buffers and slots are granted per connection. In uplink (UL) direction, however, the BS grants slots per SS and not per connection. It is the SS that decides how the UL slots are used.

There are five standardized QoS classes in WiMAX (IEEE 802.16e, to be exact). Three of them can be used for real-time connections. In unsolicited grant service (UGS), the BS allocates fixed-size grants periodically; UGS connections cannot send any bandwidth requests. In real-time polling service (rtPS), the BS periodically polls the SS by granting one slot for sending a bandwidth request, while the goal of extended real-time polling service (ertPS) is to combine the advantages of UGS and rtPS. In ertPS, the BS continues granting the same amount of bandwidth (by default, the size of this allocation corresponds to maximum sustained traffic rate of the connection) until the ertPS connection explicitly requests a change in polling size. Extended piggyback request field of the grant management subheader can be used for this purpose. If the bandwidth request size is zero, the BS may provide allocations for bandwidth request header only or nothing at all. In the latter case, contention request opportunities can be used. The two remaining QoS classes are intended for non-real time traffic. Non-real time polling service (nrtPS) is similar to rtPS except that connections are polled less frequently and they can use contention request opportunities. Best Effort (BE) connections are never polled but they can only receive resources through contention.

This paper studies scheduling in a WiMAX BS. For these purposes, we run several simulation scenarios and apply different scheduling algorithms. The rest of this paper is organized as follows: section 2 presents the proposed algorithms, sections 3 and 4 present our simulator and the simulation results, respectively, while section 5 concludes the paper.

II. SCHEDULING ALGORITHMS

As in the case of other wireless technologies, it is likely that the air interface [1, 2] will be the biggest system bottleneck with WiMAX, too. This is why the BS scheduler should make sure that the scarce resources are used effectively.

A. Deficit Round-Robin

Deficit-Round-Robin (DRR) is a well-known scheduling algorithm [3] originally developed for IP networks. In DRR, the *deficitCounter* of each active[1] connection is increased by *quantum* when the connection has its turn. If the size of the head-of-line packet of this connection is smaller than or equal to the *deficitCounter*, we send the packet and decrease the *deficitCounter* by the size of the packet. We continue sending packets as long as the *deficitCounter* allows. If the *deficitCounter* is too small for the head-of-line packet, we move to the next connection. The deficit that is stored in the

[1] Here active connection means that the connection has packets in the queue.

deficitCounter of the connection is then saved for the next round. If we were able to serve all packets of the connection, the *deficitCounter* is reset to zero.

In our WiMAX BS, however, we do not schedule packets but slots. In fact, in the UL direction we do not even know the size of the head-of-line packet. Thus, the basic DRR algorithm needs to be modified a bit for our purposes. In each frame, the virtual (UL bandwidth request) and real (DL) queue sizes are converted from bytes to slots. The number of required slots depends on the current modulation and coding scheme (MCS) of the connection. The *quantum* parameter is now given in slots. In turn, the *deficitCounter* of each connection is increased by *quantum*. Slots are granted[2] until all requests have been fulfilled or we run out of slots. Several rounds per frame can be done. If a queue drains out, the *deficitCounter* is reset to zero. It should be noted that in our variant of DRR only those DL and UL connections that are granted the last DL and UL slots of a frame can save any deficit for the next frame; all other *deficitCounters* should be zero at this stage.

B. Weighted Deficit Round-Robin

Weighted Deficit Round-Robin (WDRR) is a variant of DRR, where we adjust the *quantum* size according to connection's current MCS. We simply multiply the *quantum* by bytes per slot that the current MCS of the connection can deliver and then divide the *quantum* by six (bytes per slot for QPSK-1/2, our most robust MCS when OFDMa is used). For example, with 16QAM-3/4 we have three times bigger *quantum* than with QPSK-1/2.

WDRR might need some additional starvation-avoidance features, e.g., a coefficient that determines the final *quantum* sizes. This is for further study.

C. Proportional Fair

Proportional Fair (PF) scheduler [4] should in theory result in better throughput than the DRR scheduler, because the PF scheduler assigns slots first to those connections that have the best ratio of current achievable rate to averaged rate. In every frame, the scheduler serves the connections in this order; we repeat the following sequence as long as there are available slots or until there are no more requests.

First, find the connection with biggest *R(t)* to *T(t)* ratio and grant as many slots as the connection needs or maximum number of slots that can be allocated at a time (scheduling slot size, similar to *quantum* in DRR). *R(t)* is the number of bytes that can be sent in a single slot; it is determined by the current MCS of the connection.

Then, update the exponentially averaged rate T for all connections (including those connections that do not have any data to send in this frame) N times. N is the number of slots we just allocated. If the connection was just served:

$$T(t+1) = (1-1/t_c) * T(t) + 1/t_c * R(t). \quad (1)$$

Otherwise:

$$T(t+1) = (1-1/t_c) * T(t). \quad (2)$$

Time constant t_c is an adjustable parameter; it determines how long we can let a flow starve. In the case of 5 ms frame length and 500 slots per (DL or UL) subframe, t_c of 10000 slots corresponds to 10000*(0.005/500) = 0.1 seconds. However, if we schedule real-time connections before BE connections, the number of DL or UL slots available for the BE class cannot be known beforehand. Thus, we should take our traffic mix into account when selecting the t_c parameter. Initial value for T, *T(0)*, can be set as the expected average rate divided by the expected average number of connections.

III. WiMAX Network Simulation Model

The basic implementation of our WiMAX module is described in [5]. The module includes the following features: OFDM and OFDMa PHY levels (no MIMO[3] support yet), transport and management connections, fragmentation (a large packet can be split between several MAC PDUs), packing (several packets and packet fragments can be put into a single PDU), ranging and bandwidth request contention periods, CDMA codes for ranging and bandwidth requests, support for the most important MAC level signaling messages and the ARQ mechanism that allows retransmitting dropped PDUs.

Additionally, the module includes several different BS schedulers and it has a simple model for link adaptation. These features are described in more detail in the following sections.

A. MCS, Link Adaptation and Errors

MCS defines how many bits a connection can send in a single slot. The BS can dynamically change both the DL and UL MCS of an SS[4] based on the channel conditions. In our simulator, link adaptation is modeled by a Markov chain (see Table I), where the states represent different MCSs. Transition from a state to another is possible in each frame. We have obtained the state transition probabilities from system simulations. Naturally, these results always correspond to certain simulation parameters.

The used error rate for a 100-byte MAC PDU is 10% with each MCS when ARQ is applied and 1% when ARQ is not applied. In real life, more robust MCSs would probably be used when the connection is run without ARQ. One approach would be to use separate Markov chains for these connections. With our present model, however, less robust MCSs may be used, too.

[2] Slots are not granted immediately; here we only construct the MAP messages.

[3] Multiple Input Multiple Output.

[4] Different DL connections belonging to the same SS can use different MCSs.

TABLE I
BYTES PER SLOT VALUES AND TRANSITION PROBABILITIES OF THE LINK ADAPTATION MARKOV CHAIN

Direction	MCS	Bytes/slot	Transition probability to					
			64QAM-3/4	64QAM-2/3	16QAM-3/4	16QAM-1/2	QPSK-3/4	QPSK-1/2
DL	64QAM-3/4	27	0.79486090	0.13948325	0.06565585	0.00000000	0.00000000	0.00000000
	64QAM-2/3	24	0.28293998	0.38394350	0.33311652	0.00000000	0.00000000	0.00000000
	16QAM-3/4	18	0.01279176	0.03352881	0.87825311	0.07542632	0.00000000	0.00000000
	16QAM-1/2	12	0.00000000	0.00000000	0.02520945	0.94088229	0.03383335	0.00007491
	QPSK-3/4	9	0.00000000	0.00000000	0.00000000	0.10215911	0.78974198	0.10809891
	QPSK-1/2	6	0.00033259	0.00005309	0.00048092	0.00127569	0.01932743	0.97853028
UL	64QAM-3/4	27	0.68184019	0.21234867	0.08523002	0.01525424	0.00314770	0.00217918
	64QAM-2/3	24	0.36491229	0.41250953	0.18550725	0.02730740	0.00503432	0.00472921
	16QAM-3/4	18	0.02981995	0.06013912	0.73937498	0.13868273	0.01793856	0.01404466
	16QAM-1/2	12	0.00374267	0.00865052	0.17964833	0.67039757	0.08975355	0.04780736
	QPSK-3/4	9	0.00094722	0.00129166	0.01295961	0.11413933	0.50981658	0.36084560
	QPSK-1/2	6	0.00206103	0.00083422	0.00420929	0.01414364	0.04588231	0.93286951

B. Scheduler

The BS scheduler grants slots for the SSs according to the QoS parameters and bandwidth request sizes of the individual connections. Uplink virtual queue sizes are updated whenever slots are granted and every time a bandwidth request (that includes the real queue size) arrives. For DL connections, we use the BS queue sizes and the QoS parameters.

Our scheduler assigns slots in three stages (see Fig. 1): management connections are served first, then real-time connections, and finally non-real-time connections. Different scheduling algorithms (DRR, PF or WDRR) can be applied in the two latter stages. In DRR and WDRR, the (basic) *quantum* size is a configuration parameter[5]; a bigger *quantum* size decreases the MAP overhead, because we then serve fewer connections per frame. In PF, the t_c parameter controls the tradeoff between throughput and delay, while scheduling slot size is similar to *quantum* in DRR.

We have implemented support for three QoS classes: ertPS, rtPS and BE. ertPS and rtPS connections are served first; they are assigned slots until all ertPS and rtPS (virtual) queues are empty or until there are no slots left for this traffic. In order to avoid the starvation of BE connections, we can reserve a portion of all slots exclusively for these connections. Moreover, connection admission control should take care that there are always enough slots for the real-time connections[6]. We could easily enhance our scheduler and provide support for UGS and nrtPS classes as illustrated in Fig. 1.

UL rtPS connections are polled regularly (frequency depends on connection parameters), because they cannot participate in contention like the UL BE connections. UL ertPS connections are polled regularly, too, and during their active periods they are regularly granted slots based on their QoS requirements.

[5] Default value for quantum and scheduling slot size is 17 slots for all connections. However, we could have different values for different stages.

[6] If the connection admission control is conservative enough, it should not make a difference what scheduling algorithm we use for real-time connections – we should then be able to fulfill these requests in every frame.

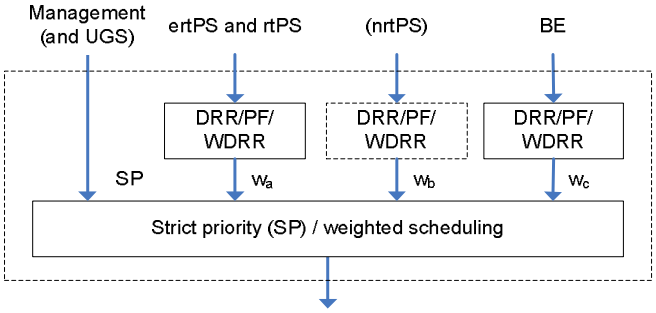

Fig. 1. Multi-stage scheduler at the BS.

In order to make sure that no bandwidth is wasted; silence suppression detection at the BS is done for UL ertPS connections: whenever an UL frame is received, a connection-specific timer is launched. When this timer expires, we go into silence state, where only polling slots are granted. We do not let the ertPS connections participate in contention, because that might introduce large medium access delays. Having a sufficient amount of request opportunities, on the other hand, would make this option more feasible. The amount of request opportunities should then be adaptive [6] as we do not want to use all of our resources for contention.

BE connections are served after ertPS and rtPS connections; they are served until the BE (virtual) queues are empty or until there are no slots left. If there are still available UL slots, they are assigned for contention.

If ARQ is enabled for a connection, we apply the following connection-internal scheduling order (at the SS, too): 1) ARQ feedback messages, 2) retransmissions and 3) all other PDUs [7]. Currently we simulate only cases with a single connection per SS.

IV. PERFORMANCE EVALUATION

We use a modified version of the ns-2 simulator [8]. The WiMAX related modifications have been described in the preceding section. Six simulations are run in each simulated case

in order to obtain 95% confidence intervals. Simulation time is always 200 seconds. One-way core network delay between a server and the BS is set to 31 ms. The only bottleneck in our system is the WiMAX air interface (see Fig. 2). The most important WiMAX network parameters are listed in Table II.

We simulate the following traffic mix: 5 VoIP connections, 5 video streaming connections (DL only); 10, 14, 18, 22, 26 or 30 web browsing connections and 5, 7, 9, 11, 13 or 15 file downloading connections per BS. All user traffic is given BE treatment except for VoIP traffic that is given rtPS treatment.

Our G.723.1 VoIP traffic source is a simple On-Off Markov model, where both On and Off period lengths are exponentially distributed with mean lengths of 1.2 s and 0.8 s, correspondingly. 24 bytes of payload is sent every 30 ms during active periods. Altogether, RTP, UDP and IP add 40 bytes of overhead, which results in a total packet size of 64 bytes. Packet header compression (from 40 bytes to 4 bytes) is applied at the BS and the SS.

TABLE II
SIMULATION PARAMETERS

Parameter	Value
PHY	OFDMa
Duplexing mode	TDD
Frame length	5 ms
Bandwidth	10 MHz
FFT size	1024
Cyclic prefix length	1/8
TTG (Transmit-receive Transition Gap)	296 PS
RTG (Receive-transmit Transition Gap)	168 PS
DL/UL permutation zone	FUSC/PUSC
DL/UL ratio	35/12
DL-MAP/UL-MAP fixed overhead	13 bytes / 8 bytes
Number of ranging opportunities	1
Ranging backoff start/end	0/15
Number of request opportunities	3
Request backoff start/end	3/15
CDMA codes for ranging and bw requests	64/192
Maximum MAC PDU size	100 bytes
Fragmentation/Packing	Yes/Yes
ARQ	For all but VoIP connections
ARQ feedback types	All
ARQ block size / window size	16 bytes / 1024
ARQ block rearrangement	No

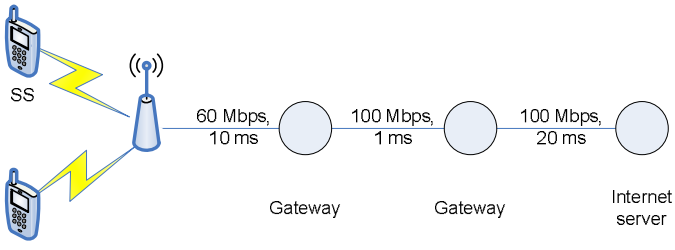

Fig. 2. Simulation topology.

The variable bit rate video traffic source is simulated according to [9]. The following parameters are used: mean rate of 125 kbps, maximum rate of 250 kbps and 1500-byte packets. UDP is used as a transport protocol and IPv4 adds 20 bytes of overhead, which results in a payload size of 1472 bytes.

Our web browsing traffic source is simplified from [10]; it is necessary to limit the variability of page sizes as the TCP goodput depends on that, too. We have a main page and 30 inline items per page, all items have a size of 4.91 kB, which results in a total page size of 152 kB (this is based on our measurements). Page reading time is uniformly distributed between 1 and 5 seconds and four NewReno TCP [11] connections are utilized. The same HTTP traffic model is used for modeling file downloading, but in that case only a single 250 kB file is downloaded. Time between two downloads is uniformly distributed between 1 and 5 seconds. A single NewReno TCP connection is utilized.

Fig. 3–11 illustrate the main results of our simulations. Both PF and WDRR perform very well (in terms of MAC throughput and TCP goodput) against DRR. This is in line with the results of [12]. Especially the good performance of WDRR is a nice surprise as this scheme should be easier to implement (and less computationally complex) than PF. The fact that PF scheduler can leave a connection without any resources for quite a long period of time (if t_c is large enough) may be a problem if ARQ timers are set to expire too soon. Moreover, sudden variations in round-trip time (RTT) might launch TCP retransmissions, and that could possibly lead into degraded TCP goodput.

WDRR outperforms PF with lower traffic loads, because the PF algorithm needs to have enough connections in order to achieve better relative throughput gain. When there are more connections, it is more likely that the PF algorithm always picks a connection with a good MCS.

When large t_c values are used in the PF scheduler, the price we have to pay for better TCP goodput is increased delay. However, Fig. 12–20 illustrate that Active Queue Management (AQM) at the BS (see [13] for details) can be used to dramatically reduce the queuing delays without sacrificing the goodput.

V. CONCLUSIONS

In this paper, we have presented a performance comparison of different schedulers for WiMAX BS. Our simulations show that PF scheduler is clearly a better choice for BE traffic than DRR scheduler. However, more studies are still needed on, e.g., the impact of different ARQ timer values.

WDRR is an interesting alternative to PF as it is somewhat simpler in implementation. As WDRR seems to outperform PF when the number of connections is low, it could be feasible to combine PF and WDRR so that there would be a certain threshold (e.g., the number of users or connections) after which the scheduling algorithm would change from WDRR to PF.

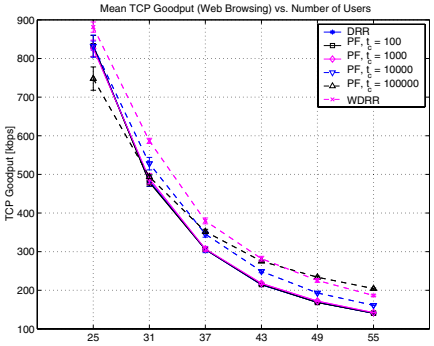

Fig. 3. TCP goodput for web browsing.

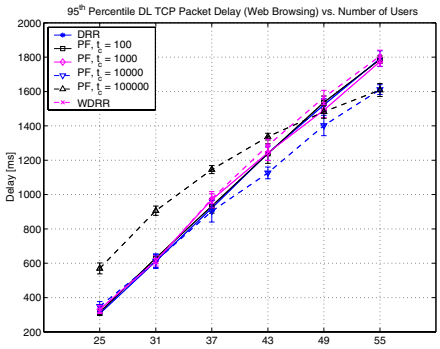

Fig. 4. 95th percentile DL TCP packet delay for web browsing.

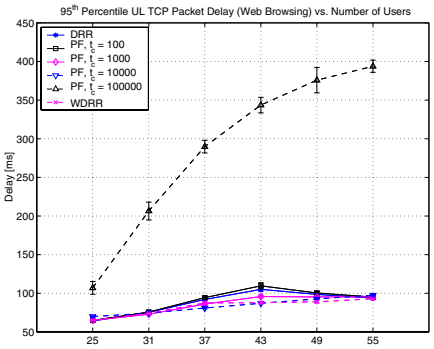

Fig. 5. 95th percentile UL TCP packet delay for web browsing.

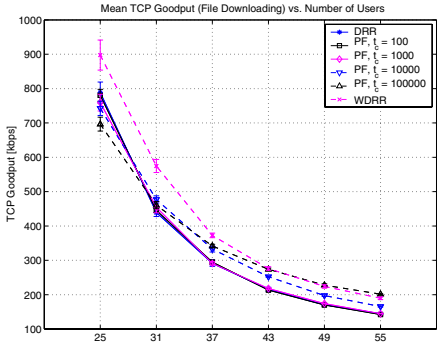

Fig. 6. TCP goodput for file downloading.

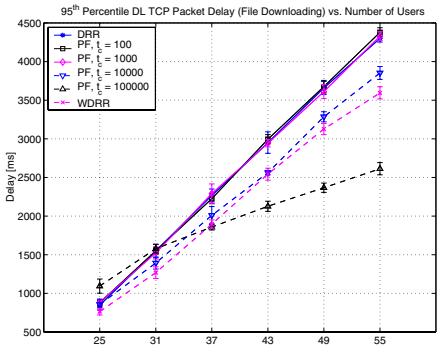

Fig. 7. 95th percentile DL TCP packet delay for file downloading.

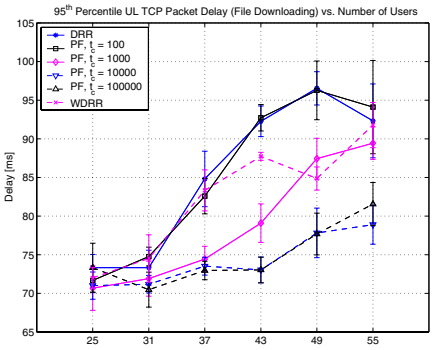

Fig. 8. 95th percentile UL TCP packet delay for file downloading.

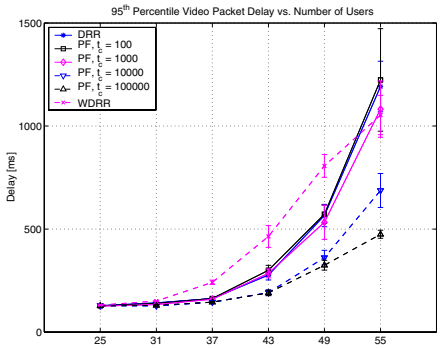

Fig. 9. 95th percentile DL video packet delay.

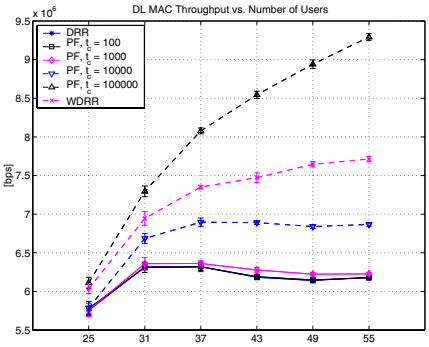

Fig. 10. DL MAC throughput.

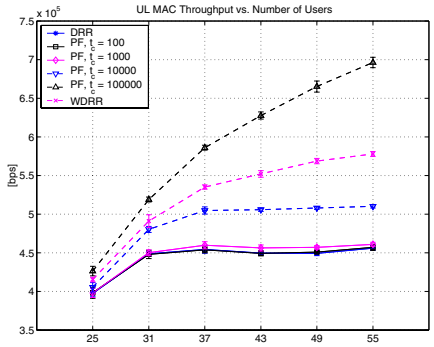

Fig. 11. UL MAC throughput.

For VoIP and other real-time traffic, DRR is still the best choice. It is not acceptable to let VoIP connections starve every now and then (when the most robust MCSs are used) just because that would lead into better MAC throughput. In fact, with PF scheduling, VoIP delay could grow intolerable if the number of VoIP connections is significant.

ACKNOWLEDGMENT

The authors would like to thank everyone involved in the ns-2 simulator development at the Telecommunication laboratory of University of Jyväskylä.

REFERENCES

[1] Air interface for fixed broadband wireless access systems. IEEE Standard 802.16, Jun. 2004.

[2] Air interface for fixed broadband wireless access systems – amendment for physical and medium access control layers for combined fixed and mobile operation in licensed bands. IEEE Standard 802.16e, Dec. 2005.

[3] M. Shreedhar and G. Varghese, "Efficient fair queueing using Deficit Round-Robin," IEEE/ACM Transactions on Networking, vol. 4, pp. 375-385, Jun. 1996.

[4] A. Jalali, R. Padovani, and R. Pankaj, "Data Throughput of CDMA-HDR: A High Efficiency-High Data Rate Personal Communication Wireless System," Proceedings of the IEEE VTC2000-Spring, Tokyo, Japan, May 2000.

[5] A. Sayenko, O. Alanen, J. Karhula and T. Hämäläinen," Ensuring the QoS Requirements in 802.16 Scheduling," Proceedings of IEEE/ACM MSWiM 2006, Torremolinos, Spain, Oct. 2006.

[6] A. Sayenko, O. Alanen and T. Hämäläinen, "Adaptive Contention Resolution Parameters for the IEEE 802.16 Networks," Proceedings of Qshine 2007, Vancouver, Canada, Aug. 2007.

[7] A. Sayenko, V. Tykhomyrov, H. Martikainen and O. Alanen, "Performance Analysis of the 802.16 ARQ Mechanism," Proceedings of IEEE/ACM MSWiM 2007, Chania, Greece, Oct. 2007.

[8] UCB/LBNL/VINT, "Network Simulator – ns (version 2)," Jun. 2007. http://www.isi.edu/nsnam/ns/index.html

[9] B. Maglaris, D. Anastassiou, P. Sen, G. Karlsson and J. Robbins, "Performance Models of Statistical Multiplexing in Packet Video Communications," IEEE Transactions on Communications, vol. 36, pp. 834-844, Jul. 1988.

[10] M. Molina, P. Castelli and G. Foddis, "Web Traffic Modeling Exploiting TCP Connections' Temporal Clustering through HTML-REDUCE," IEEE Network, vol. 12, pp. 46-55, May-Jun. 2000.

[11] S. Floyd and T. Henderson, "The NewReno Modification to TCP's Fast Recovery Algorithm," RFC 2582, Apr. 1999.

[12] Chingyao Huang, Hung-Hui Juan, Meng-Shiang Lin and Chung-Ju Chang, "Radio Resource Management of Heterogeneous Services in Mobile WiMAX systems," IEEE Wireless Communications, vol.14, pp.20-26, Feb. 2007.

[13] J. Lakkakorpi, A. Sayenko, J. Karhula, O. Alanen and J. Moilanen, "Active Queue Management for Reducing Downlink Delays in WiMAX," Proceedings of IEEE VTC2007-Fall, Baltimore, MD, USA, Oct. 2007.

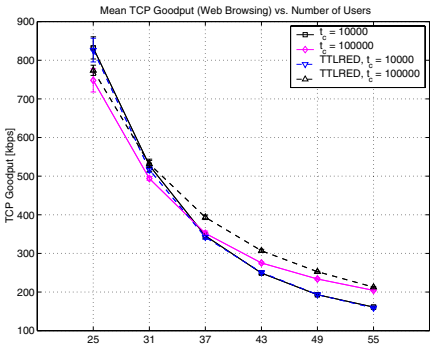

Fig. 12. TCP goodput for web browsing with/without DL AQM.

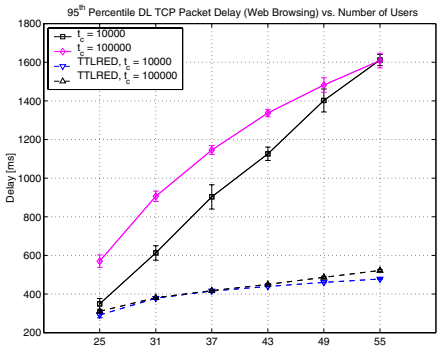

Fig. 13. 95th percentile DL TCP packet delay for web browsing with/without DL AQM.

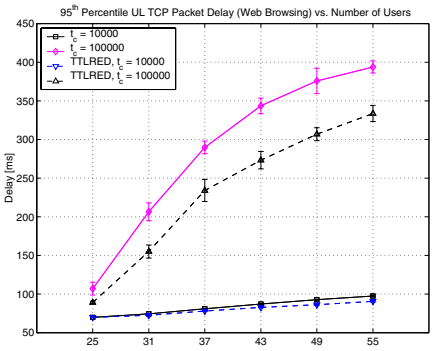

Fig. 14. 95th percentile UL TCP packet delay for web browsing with/without DL AQM.

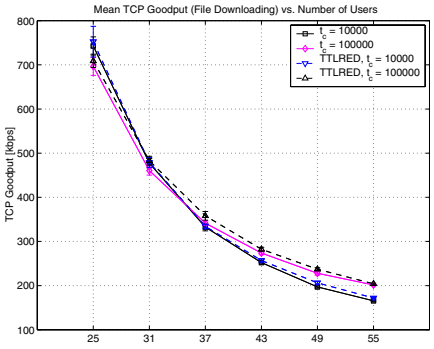

Fig. 15. TCP goodput for file downloading with/without DL AQM.

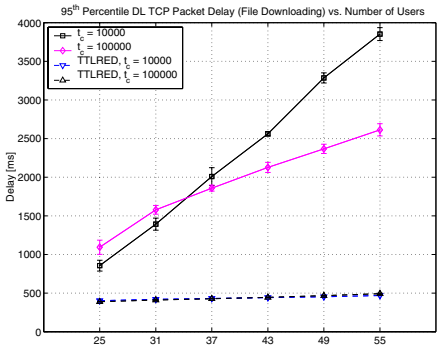

Fig. 16. 95th percentile DL TCP packet delay for file downloading with/without DL AQM.

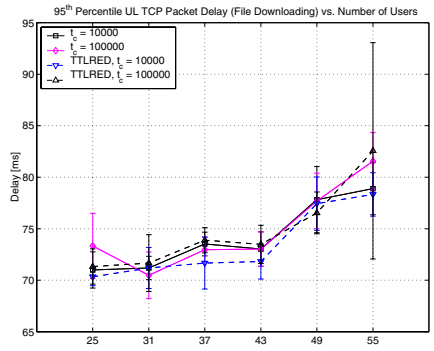

Fig. 17. 95th percentile UL TCP packet delay for file downloading with/without DL AQM.

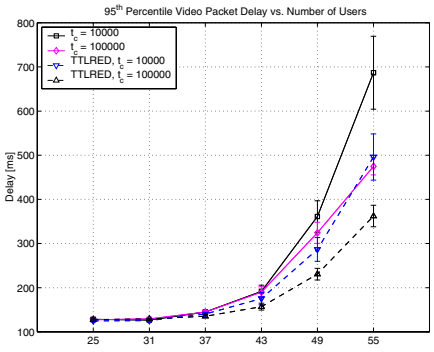

Fig. 18. 95th percentile DL video packet delay with/without DL AQM.

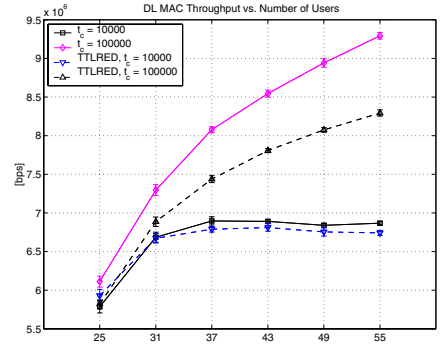

Fig. 19. DL MAC throughput with/without DL AQM.

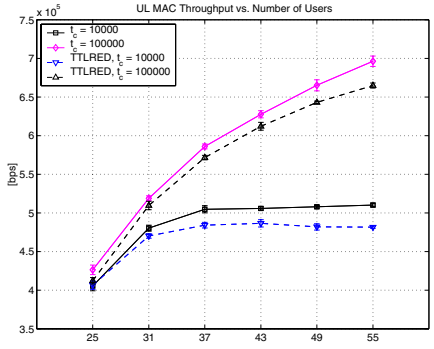

Fig. 20. UL MAC throughput with/without DL AQM.