QoS guarantee over DQCA for Wireless LANs with heterogeneous traffic

Abstract: Distributed Queuing Collision Avoidance (DQCA) is an efficient MAC protocol designed for infrastructure Wireless LANs. In this paper, four algorithms are proposed that alter the FIFO scheduling order of DQCA in order to meet specific network requirements. The proposed schemes combine the efficiency of opportunistic scheduling with the QoS provisioning through service differentiation. The throughput and delay performance has been evaluated through simulations for a scenario with heterogeneous traffic.

Keywords: QoS, Cross-Layer, DQCA, NGN, MAC

1. Introduction

Wireless Local Area Network (WLAN) technology has become a very appealing solution to network connectivity, providing user mobility, flexibility and easy deployment at a relatively low cost. As the popularity of WLANs grows, the need for higher transmission rates and Quality of Service (QoS) guarantees becomes imperative, especially for real-time multimedia applications.

The process of selecting the optimal transmission rate depending on the channel condition is known as link adaptation and is usually performed at the Medium Access Control (MAC) layer. The time-varying nature of the wireless channel and the multirate PHY capability can be further exploited by the MAC layer to achieve more efficient scheduling. The key idea is to encourage communication between nodes with good channel condition, as reflected by the perceived SNR at the receiver. Although the consideration of a PHY layer metric, such as the SNR, in MAC layer scheduling decisions is a violation of the OSI architecture, it can be exploited to optimize the performance of the system. This exchange of information between different OSI layers is known as Cross-Layer design and is a very promising field of investigation [1]. Several related schemes have been proposed in the literature [2]-[4]. Although opportunistic scheduling can increase the capacity of the system, it cannot provide QoS guarantees that are essential to real-time traffic such as video and voice applications. A common approach to alleviate this problem is to incorporate QoS requirements into the scheduling decisions taken at the MAC layer. The relatively new IEEE 802.11e standard addresses this issue by providing service differentiation [5].

In this context, we propose a number of scheduling algorithms that combine the efficiency of cross-layer opportunistic schemes with QoS provisioning through service differentiation. Our research is based on a near-optimum MAC protocol named DQCA (Distributed Queuing Collision Avoidance) [6]. DQCA is part of an extended family of distributed protocols, some of which are being commercially deployed. In particular, Ether2 [7] is currently implementing a distributed protocol of the same family for Ethernet together with a DQCA-based wireless air-interface for WLANs.

This paper extends and completes our preliminary work presented in [8], that focused on a scenario with voice and data traffic. In this paper, we extend our study to four scheduling algorithms and examine in detail the effect of opportunistic and service differentiation policies on the system performance. The objective of this contribution is to

This work has been funded by the NEWCOM++ Research Project.

investigate the impact of different cross-layer QoS-aware scheduling on the performance of DQCA-based networks under a realistic, heterogeneous traffic assumption.

The rest of the paper is organized as follows. An overview of the DQCA protocol is given in Section 2 and the proposed scheduling algorithms are presented in Section 3. Section 4 describes the study case scenario and simulation results are provided in Section 5. Finally, Section 6 is devoted to conclusions.

2. THE DQCA PROTOCOL

The purpose of this section is to highlight the basic features of DQCA which are essential for the understanding of the proposed scheduling algorithms. A detailed explanation of DQCA, along with the protocol operating rules, can be found in [6].

DQCA is a distributed high-performance MAC that behaves as a random access mechanism under low traffic conditions and switches smoothly and automatically to a reservation scheme when traffic load grows. As demonstrated in [6], DQCA outperforms the DCF MAC mechanism implemented in the 802.11 standard and remains stable even when the traffic load exceeds the channel capacity.

DQCA is a MAC protocol designed for an infrastructure WLAN in which a number of nodes share the wireless channel in order to communicate with an Access Point (AP). The DQCA frame can be divided in three parts, the Contention Window (CW), the data slot and the feedback part, as illustrated in Figure 1. The CW is further divided in *m* control slots in which the nodes may send a short chip sequence named Access Request Sequence (ARS) to reserve the channel. The ARS does not contain any information but has a specific, predefined pattern that enables collision detection [9]. The data slot is reserved for the transmission of data packets. In the third part, the AP broadcasts a Feedback packet (FBP) that contains the data acknowledgment and the states of the *m* control slots. It also announces the completion of a data message transmission and can contain additional information if required. An error-free reception of the FPB by every node at the end of each frame has been assumed. Finally, a Short Interframe Space (SIFS) interval is introduced before and after the transmission of the FBP.

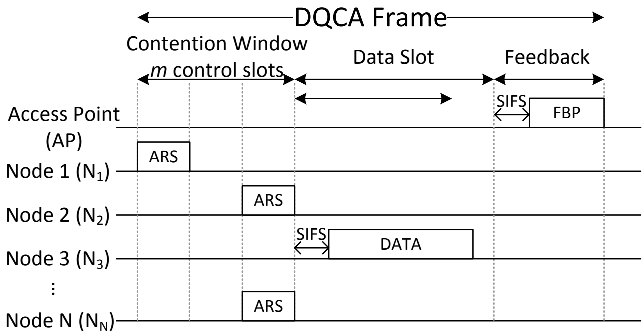

Figure 1 – DQCA Frame Structure

The protocol is based on two concatenated distributed queues, the Collision Resolution Queue (CRQ) and the Data Transmission Queue (DTQ). The CRQ is responsible for the resolution of collisions among ARS and the DTQ handles the transmission of data. The number of occupied positions (or elements) in each queue is represented by an integer counter (*RQ* and *TQ* for the CRQ and the DTQ, respectively). Both counters have the same value for all the nodes in the system and are updated according to a set of rules at the end of each frame. Each node must also maintain and update another set of counters that reveal its position in the queue (*pRQ* and *pTQ* for the CRQ and the DTQ, respectively). By the term "position" it is meant the relative order of arrival (or age) of the node in the respective queue. In the CRQ, each position (or element) is occupied by a set of nodes that suffered an ARS collision (i.e. attempted an ARS transmission in the same control slot of the same

CW). The DTQ contains the nodes that successfully reserved the channel through an ARS and therefore each queue element corresponds to exactly one node.

3. Cross-Layer Scheduling Algorithms

In DQCA, the data transmissions take place in a FIFO (First-In First-Out) order, thus ensuring fairness among nodes. In this section, we present four algorithms that modify the scheduling discipline of the DTQ. The main idea is on one hand to exploit the multirate capability of the PHY layer by opportunistically transmitting when higher rates are available and on the other hand to meet QoS requirements through service differentiation.

To implement the proposed algorithms some modifications need to be made to the DQCA protocol. First, a link adaptation mechanism must be incorporated in order to adapt the transmission rate to the channel condition. According to this scheme, the AP measures the SNR of the link whenever an ARS is successfully received and selects the maximum rate that can be supported for upcoming data transmissions from the respective node. This rate is included in the FBP so that the node can appropriately adjust its modulation and coding scheme at the time of data transmission. Obviously, when the link condition is good, a modulation and coding set of higher order can be employed to yield higher rates.

Another issue is that in order to perform QoS oriented scheduling, service differentiation capabilities should be introduced to the DQCA protocol. To this end, all traffic flows are classified into P service classes ($P \geq 1$), with class P having the most demanding requirements, usually translated to most stringent delay constraints. It has been assumed that it is possible to form P distinguishable types of ARS, one for each service class. Each node that wants to access the channel must transmit an ARS of the appropriate format, so as to make the service class of its traffic load known to the other nodes of the network.

3.1 – Cross-Layer Opportunistic Scheduling Algorithm (CL-algorithm)

The first scheme, named CL-algorithm, implements a distributed opportunistic policy that prioritizes the node with the best channel quality, irrespectively of its service class. The AP keeps a table with the available rates of all the nodes that are waiting to transmit in the DTQ. These rates have been acquired through link estimation for every received ARS. The entries of this table are fed back to the nodes in the form of a vector that is included in the FBP. This vector has a specific structure; it contains the rates of all the nodes in the DTQ placed by ascending order of their respective *pTQ* values, that is also an indication of their waiting time in the queue. Upon receipt of the FPB, the nodes sort the entries of the vector from higher to lower rate and the node with the highest rate gains access to the channel. If more than one node has the same rate, the one with the smallest *pTQ* value has priority.

3.2 – Strict Service Priority Scheduling Algorithm (SP-algorithm)

The second scheme, called SP-algorithm, implements a strict service priority policy, ensuring that the nodes of the highest-priority service class (P) always transmit first. As a result, delay-sensitive applications are more likely to meet their QoS requirements.

This scheme could be better understood by visualizing P transmission queues, one for each service class, instead of a single DTQ. Each queue has its own counter, denoted by TQ_i ($0<i\leq P$), that indicates the number of nodes in the queue. A node can belong to only one queue at a time and its queue position is expressed, as before, by the *pTQ* value. Note that this algorithm does not require additional overhead, provided that distinguishable types of ARS are available to reveal the service class of each message. This is essential, since at the end of the frame when the protocol rules are executed, the nodes must know the service class of any successful ARS in order to update the corresponding TQ_i counter.

3.3 –Cross-Layer with Strict Service Priority Scheduling Algorithm (CLSP-algorithm)

The CLSP-algorithm is a combination of the two previously described schemes. It implements a strict priority policy among service classes and an opportunistic scheduling among the nodes of the same service class. Similarly to the CL-algorithm, a rate vector must be transmitted, however in this case the entries of the vector are sorted by order of the service class and the position of the nodes in each service class queue.

3.4 – Virtual Priority Function Scheduling Algorithm (VPF-algorithm)

The fourth technique employs a single DTQ, but now the transmission order is determined with the use of a virtual priority (*VPF*) function that has the general form

$$VPF = f\left(PHY\ Parameters,\ MAC\ parameters,\ Service\ \mathrm{Class}\right) \qquad (1)$$

The *VPF* is known to all nodes and the AP is responsible for providing the required feedback so that at the end of each frame every node is capable of calculating the *VPF* values of all the nodes in the DTQ, including its own. The node with the highest *VPF* value is then enabled to transmit. If more than one node has the same *VPF* value, then priority is given to the one with the longest waiting time in the DTQ (i.e. smallest *pTQ* value). To evaluate the VPF-algorithm, the following definition has been chosen:

$$VPF = \alpha \cdot \left(\frac{service\ class\ id}{P}\right) + \left(1-\alpha\right) \cdot \left(\frac{R}{R_{\max}}\right) \cdot \frac{1}{pTQ} \qquad (2)$$

Where α is a tuneable weighting factor ($0 \le \alpha \le 1$), *P* is the total number of service classes, *Service class id* indicates the service class ([1,*P*]), *R* is the transmission rate and *pTQ* is an integer within [1,*TQ*], where *TQ* is the total number of nodes in the DTQ. Note that the VPF has two parts. The first part depends solely on the service class whereas the second part is proportional to the (normalized) transmission rate divided by *pTQ*, meaning that preference is given to nodes with higher rates and longer waiting times in the DTQ.

4. Case Study

The proposed algorithms have been evaluated through simulations performed by a C++ link-layer simulation tool. The basic features of the simulated scenario are described next.

4.1 – Channel Model

Without loss of generality, the underlying PHY layer follows the IEEE 802.11g specification [10]. A transmission set of eight rates $R = \{6, 9, 12, 18, 24, 35, 48, 54\}\ Mbps$ is assumed, with higher rates requiring better channel conditions. Following the methodology in [11], the wireless channel has been modelled as an eight-state discrete Markov chain where each state represents one of the eight available rates. The transition matrix *T* used in the simulation scenario, without losing generality, is given in Table I. The coherence time of the channel is supposed to be long enough to ensure that the channel state remains unchanged from the moment of the SNR measurement until the data transmission.

4.2 – Traffic Models

The simulated scenario includes three types of data sources that are described below.

4.2.1 – Data Traffic Sources

The data traffic sources have a Poisson distributed generation process with arrival rate λ. The size of the generated messages is exponentially distributed with a mean value equal to

$10 \cdot L$, with L set to 1000 bytes. This results to an offered load of C= $\lambda \cdot (10 \cdot L) \cdot 8 \cdot 10^{-6}$ Mbps. Since data messages are relatively large, their transmission takes place in consecutive DQCA frames with L being the maximum number of bytes transmitted per each frame.

4.2.2 – Voice Traffic Sources

The voice traffic sources are based on the Brady's ON-OFF model. The time spent at the ON and OFF state is exponentially distributed with mean values of 1s and 1.35s respectively. While at the ON state, packets of 160 bytes are generated every 20ms resulting to a CBR of 64 kbps and a mean offered load of 27.23 kbps, as defined in the G.711 voice codec [12]. The packets generated within a single ON interval will be referred to as a voice burst. Since voice packets are relatively small, voice nodes transmit all the buffered packets of the same burst in a single DQCA frame, once they gain channel access.

4.2.3 – Video Traffic Sources

A near real-time video model defined in [13] has been used for the generation of streaming video traffic. Each video frame arrives at a regular time interval, defined by a frame per second rate of 10. Each frame consist of 25 packets whose size follows a truncated Pareto distribution (α=1.2, Min = 50 bytes and Max = 200 bytes). Due to the video encoding process, there is an arrival delay between the packets of the same frame that is also modelled by a truncated Pareto distribution (α=1.2, Min = 2.5 ms and Max = 4 ms). The mean offered load for video traffic is 180 kbps. Similarly to voice nodes, video nodes transmit all the buffered packets that belong to the same frame in a single DQCA frame.

4.3 – Service Differentiation

Four service classes are defined for Background, Best-Effort, Video and Voice traffic, with service class ids from 1 to 4, respectively, following the paradigm used in IEEE 802.11e [5]. Voice traffic is assigned the highest priority due to stringent delay constraints. A maximum delay of 150 ms per voice packet is tolerated, after which the packet is dropped. For the video class, the maximum delay has been set to 300 ms per packet. Best-effort data can tolerate a delay up to 5 s and finally background data has no delay constraints.

Table I- Channel Transition Matrix

Channel Transition Matrix

$$
T = \begin{array}{c}
\overbrace{\begin{array}{cccccccc} 6 & 9 & 12 & 18 & 24 & 36 & 48 & 54 \end{array}}^{\text{future state}} \\
\begin{bmatrix}
0.4 & 0.5 & 0.1 & 0 & 0 & 0 & 0 & 0 \\
0.1 & 0.4 & 0.5 & 0 & 0 & 0 & 0 & 0 \\
0 & 0.1 & 0.4 & 0.4 & 0.1 & 0 & 0 & 0 \\
0 & 0 & 0.1 & 0.4 & 0.4 & 0.1 & 0 & 0 \\
0 & 0 & 0 & 0.1 & 0.5 & 0.4 & 0 & 0 \\
0 & 0 & 0 & 0 & 0.3 & 0.5 & 0.2 & 0 \\
0 & 0 & 0 & 0 & 0.1 & 0.2 & 0.5 & 0.2 \\
0 & 0 & 0 & 0 & 0 & 0.1 & 0.4 & 0.5
\end{bmatrix}
\end{array}
\left.\begin{array}{c} 6 \\ 9 \\ 12 \\ 18 \\ 24 \\ 36 \\ 48 \\ 54 \end{array}\right\} \begin{array}{c}\text{current} \\ \text{state}\end{array}
$$

Table II – PHY/MAC Layer Parameters

Parameter	Value
MAC Header	34 bytes
PHY Header	20 µs
SIFS	10 µs
Propagation Time	1 µs
Number of control slots m	3
ARS	10 µs
FBP	13 bytes + CL_Overhead
CL_Overhead	
DQCA and SP-algorithm	-
CL,CLSP,VPF-algorithms	$3 \cdot TQ$ bits

4.4 – MAC Layer Parameters

The DQCA protocol parameters are summarized in Table II. Control frames are transmitted at the minimum rate of 6 Mbps to ensure correct reception. The FBP has a length of 13 bytes, including 2 bytes for the Frame Control (FC) field, 6 bytes for the feedback information, 1 byte for the ACK and 4 bytes for the FCS (Frame Control Sequence). As described in Section 3, some of the proposed algorithms require a rate vector with the rates of all nodes in the DTQ to be incorporated in the FBP. A set of eight rates can be fully represented using 3 bits. If TQ is the number of nodes in the DTQ, then $3 \cdot TQ$ bits must be appended to the FBP. Note that when multiple transmission queues are defined, as in the case of the CLSP-algorithm, the number of nodes is calculated as $TQ = \sum_{i=1}^{P} TQ_i$.

5. Performance Evaluation

A simulation-based throughput and delay performance evaluation of the proposed algorithms with respect to DQCA is given in this section. A more thorough comparison with other MAC schemes, such as the 802.11e protocol, although interesting, is not included in this paper due to space constraints. Throughput is defined as the total number of successfully transmitted data bits to the unit of time. The relative throughput is defined as the ratio of the successfully transmitted bits to the number of generated bits and is evaluated for each class as well as for the whole system. The mean delay is defined as the average time from the generation of a packet until its successful transmission.

In the considered scenario each node generates a single type of traffic, so the terms node and traffic flow (tr. flow) can be used interchangeably. Initially, a constant number of 10 nodes have been considered that generate background traffic of 1.8 Mbps each. Then, tr. flows of voice, video and best-effort service classes are gradually added. The number of additional tr. flows is marked on the x-axis of all the figures presented in this section. For example, at x=0 the offered load consists of the 10 background tr. flows. At x=1, three tr. flows are added, a voice, a video and a best-effort traffic source, thus raising the total number of nodes to 13. At the final evaluation point (x=20) the system contains 70 nodes, 10 with background traffic and 20 with each of the other three service classes. Note that the traffic generated is approximately 27.23 kbps per voice node, 180 kbps per video node and 1 Mbps per best-effort node. Hence, at each step the offered load increases by about 1.2 Mbps. The approximate value of total offered load is marked at the upper part of the figures. Note also that for the VPF-algorithm the parameter α has been set to 0.5.

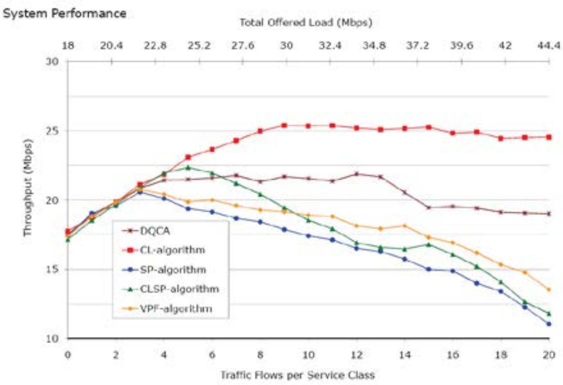

Figure 2 – Total Throughput Performance

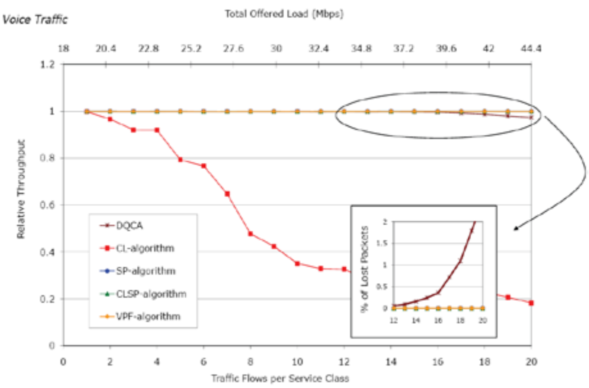

Figure 3 – Relative Throughput for Voice Class

The total system throughput is illustrated in Figure 2. The best performance is achieved by the CL-algorithm since, due to the opportunistic scheduling, higher transmission rates are used. A maximum throughput of 25 Mbps is reached and maintained even when the traffic load increases. The performance of DQCA comes next with a maximum throughput of approximately 21.5 Mbps, which drops after point 14 and stabilizes at 19Mbps. The other three proposed algorithms yield a lower throughput that decreases as the traffic load grows. The CLSP-algorithm performs better than the SP-algorithm especially under light traffic (e.g. for 5 tr. flows CLSP achieves 22.3Mbps which is about 3Mbps higher than the SP-algorithm and 1Mbps higher than DQCA). The VPF-algorithm performs slightly better than the SP-algorithm for light traffic and has a milder decrease rate. As a result it eventually outperforms both SL and CLSP algorithms for higher traffic loads.

In order to appreciate the contribution of the proposed schemes, the relative throughput per service category has been plotted. The SP, CLSP and VPF algorithms that employ service differentiation achieve a relative throughput of 1 for the high-priority voice class (Figure 3). DQCA also has a close to one throughput that slightly drops as the traffic grows. The percentage of lost packets due to excess delay is shown at the rightmost part of Figure 3 and for DQCA it remains under 1% for up to 17 voice tr. flows. On the contrary, the CL-algorithm totally fails to satisfy the voice QoS requirements.

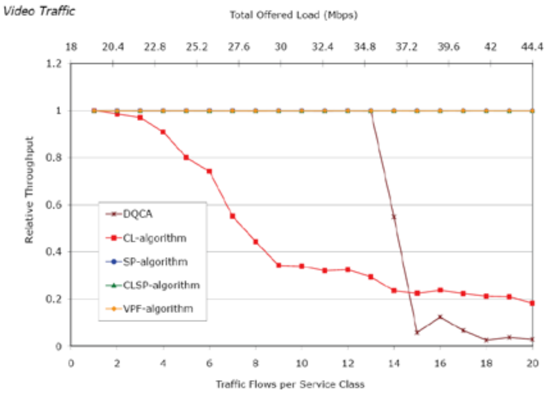

Figure 5 – Relative Throughput for Video Class

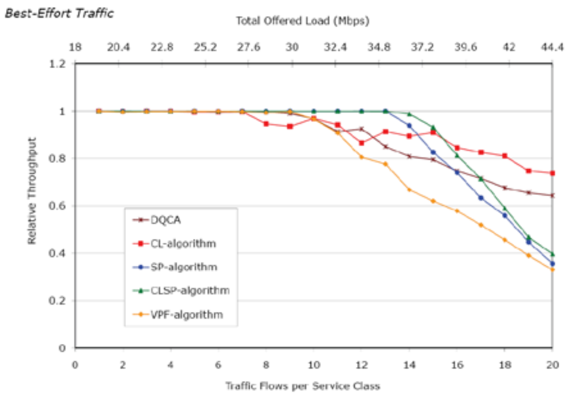

Figure 4 – Relative Throughput for Best-effort Class

In the case of video traffic (Figure 5), SP, CLSP and VPF algorithms again guarantee a relative throughput of 1. DQCA can successfully support up to 13 video nodes, but after that point the video performance drops dramatically. Best-effort traffic has very relaxed QoS constraints and therefore the CL-algorithm yields the highest throughput performance (Figure 4). The strict priority schemes (SP and CLSP-algorithms) have a throughput of 1 for up to 13 best-effort nodes and then deteriorate rapidly. The mild-priority VPF-algorithm can fully support 9 best-effort flows but the drop in performance is not so steep. Finally, for the background traffic (Figure 6), the CL-algorithm performs better with a clear difference, DQCA is second and the VPF-algorithm comes third. The strict priority schemes fail to deliver any background traffic beyond point 13 (that corresponds to a total of 49 nodes).

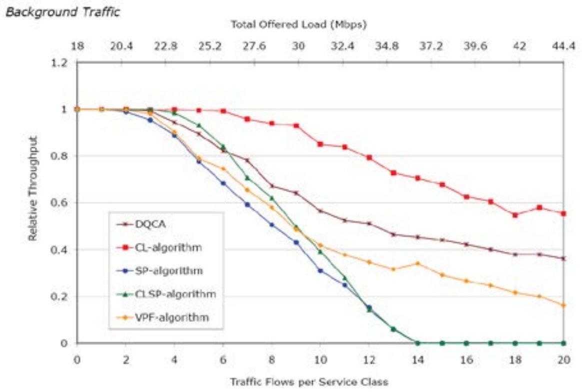

Figure 6- Relative Throughput for the Background Class

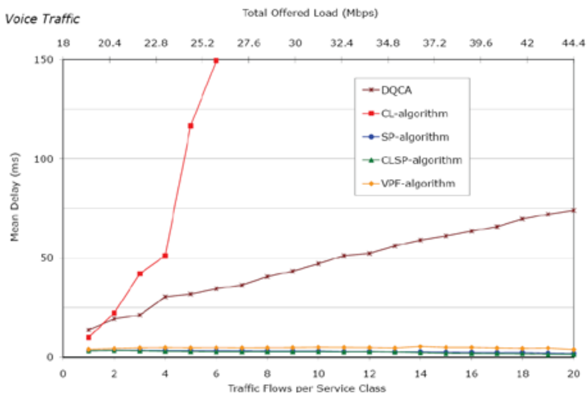

Figure 7 – Mean Delay for the Voice Class

The above observations make sense since all algorithms have the same bandwidth but allocate resources in a different way. Hence, there is a trade-off between fairness among all service classes and QoS provisioning. As the traffic load grows, the majority of packets transmitted under the service priority schemes belong to the voice and video classes. Since those packets are generally small, the impact of the PHY/MAC overhead is more noticeable and is reflected as a decrease on the total system throughput.

The delay performance per service class has been plotted in Figure 7 and Figure 8. For the SP, CLSP and VPF algorithms that prioritize voice the mean voice delay is kept below 5ms. The mean voice delay in DQCA gradually increases as the traffic grows but is kept below 75ms. On the contrary, the delay for the CL-algorithm quickly exceeds 150ms, which is the maximum tolerated value by the voice service. Similar behaviour can be observed in Figure 8 for the video service where the maximum tolerated delay is 300ms.

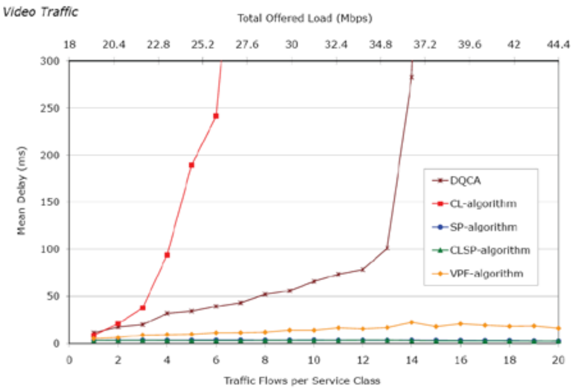

Figure 8 – Mean Delay for the Video Class

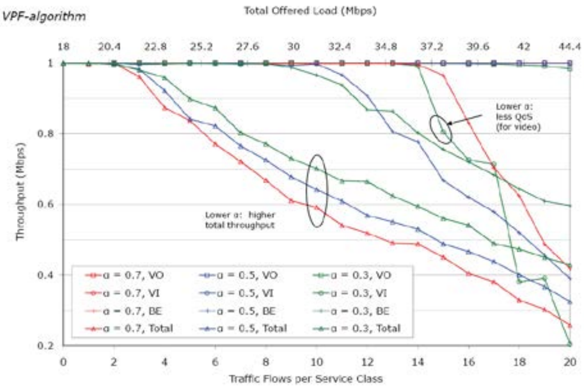

Figure 9 – Relative Throughput of the VPF-algorithm

Finally, Figure 9 depicts the throughput achieved by the VPF-algorithm for three values of the parameter α (α=0.3, 0.5 and 0.7). Smaller values of α correspond to a more opportunistic scheduling scheme whereas larger values mean that service differentiation plays a more important role. The throughput for the voice, video and best-effort classes has been plotted, as well as the total throughput of the system. It is evident that more opportunistic schemes have a higher total throughput but cannot always guarantee QoS.

6. Conclusions

In this paper, four scheduling algorithms based on an efficient distributed MAC protocol named DQCA have been proposed. The CL-algorithm implements an opportunistic transmission scheme that maximizes the system throughput but does not perform so well when considering the specific QoS requirements. The SP-algorithm employs strict service differentiation that guarantees high performance to the delay-sensitive classes of voice and video at the expense of lower throughput for best-effort and background classes. The CLSP-algorithm is a combination of the above schemes and provides QoS provisioning along with an enhanced throughput and delay performance. Finally, the VPF-algorithm is a compromise option between opportunistic scheduling and service differentiation and its performance depends on the selection of the tuning parameter α, which could be dynamically adjusted to adapt the system performance to the QoS requirements.

References

[1] S. Shakkottai, T. Rappaport, and P. Karlsson. Cross-layer design for wireless networks. Communications Magazine, IEEE, 41(10):74–80, Oct 2003.

[2] B. Sadeghi, V. Kanodia, A. Sabharwal, and E. Knightly. Opportunistic media access for multirate ad hoc networks. In Proceedings of the 8th MobiCom ’02, pages 24–35, New York, NY, USA, 2002. ACM.

[3] S. Bouam and J. Othman. A 802.11 multiservices cross-layer approach for QoS management. Vehicular Technology Conference, 2004. VTC2004-Fall. 2004 IEEE 60th, 4:2698– 2702 Vol. 4, Sept. 2004.

[4] J. Wang, H. Zhai, and Y. Fang. Opportunistic packet scheduling and media access control for wireless LANs and multi-hop ad hoc networks. Wireless Communications and Networking Conference, 2004. WCNC. 2004 IEEE, 2:1234–1239 Vol.2, March 2004.

[5] IEEE Std. 802.11e, “Medium Access Control (MAC) Quality of Service Enhancements”, 2005.

[6] J. Alonso-Zárate, C. Verikoukis, E. Kartsakli, A. Cateura, and L. Alonso. A near-optimum cross-layered distributed queuing protocol for wireless LAN. Wireless Communications, IEEE, 15(1):48–55, February 2008.

[7] www.ether2.com, “Ether 2 -Distributing the Queue”.

[8] E. Kartsakli, A. Cateura, J. Alonso-Zárate, C. Verikoukis, and L. Alonso. Cross-layer enhancement for wlan systems with heterogeneous traffic based on DQCA. ICC ’07, June 2007.

[9] G. M. Campbell and W. Xu, “Method and apparatus for detecting collisions on and controlling access to a transmission channel,” United States Patent 6292493, September 2001.

[10] IEEE Std. 802.11g, “Further Higher Data Rate Extension in the 2.4 GHz Band,” 2003.

[11] A. Konrad, B. Y. Zhao, A. D. Joseph, and R. Ludwig. A markov-based channel model algorithm for wireless networks. In Proceedings of the 4th ACM MSWIM ’01, pages 28–36, New York, NY, USA, 2001. ACM.

[12] “Pulse code modulation (PCM) of voice frequencies,” ITU-T Recommendation G.711, November 1998.

[13] 3GPP2, TSG-C.R1002-0, “cdma2000 Evaluation Methodology Revision 0,” 3GPP2, December 2004.

[14] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, “RTP: A Transport Protocol for Real-Time Applications,” RFC 1889, IETF Audio-Video Transport Working Group, January 1996.